TE
662
.A3
no.
FHWA-
RD-
73-44

eport No. FHWA-RD-73-44

REDICTION OF LONG-TERM STRESS RANGES
ser's Manual–Bridge Dynamic Stress Analysis
(BRGSTRS)

DEPARTMENT OF
TRANSPORTATION
JUL 24 1974
LIBRARY

J.W. Fothergill, H.Y. Lee, and P.A. Fothergill

June 1973
Final Report

This document is available to the public
through the National Technical Information
Service, Springfield, Virginia 22151

Prepared for
FEDERAL HIGHWAY ADMINISTRATION
Offices of Research & Development
Washington, D.C. 20590

NOTICE

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

The contents of this report reflect the views of the contracting organization, which is responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policy of the Department of Transportation. This report does not constitute a standard, specification, or regulation.

TE
662
.A3
no.
FHWA-
RD-
73-44

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. FHWA-RD-73-44	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle
PREDICTION OF LONG-TERM STRESS RANGES
User's Manual - Bridge Dynamic Stress Analysis

5. Report Date
June 1973

6. Performing Organization Code

7. Author(s)
J. W. Fothergill, H. Y. Lee, and P. A. Fothergill

8. Performing Organization Report No.

9. Performing Organization Name and Address
Integrated Systems, Inc.
6900 Wisconsin Avenue
ChevyChase, Maryland 20015

10. Work Unit No.
FCP 35G1-012

11. Contract or Grant No.
DOT-FH-11-7904

12. Sponsoring Agency Name and Address
Offices of Research and Development
Federal Highway Administration
U. S. Department of Transportation
Washington, D. C. 20590

U.S.

13. Type of Report and Period Covered
Final Report

14. Sponsoring Agency Code

15. Supplementary Notes
FHWA Contract manager: W. L. Armstrong (HRS-11)

16. Abstract

General: The development of a computer simulation program system for generating highway traffic crossing a bridge and calculating the resulting stresses and stress ranges is described. The total system consists of four stand-alone programs which can be used either independently or as a dependent series where output from one program is used as input to the next. Output consists of loading and stress range histograms.

Report 73-44: A description of the dynamic stress analysis program, BRGSTRS, is given. Included are utilization instructions, data preparation instructions, output and variables definitions, and a sample case.
This report is third in a series. The others in the series are:

FHWA No.	Short Title
FHWA-RD-73-42	Study Report
FHWA-RD-73-43	User's Manual - Bridge Load Generator
FHWA-RD-73-45	User's Manual - Stress Histogram Prediction System

DEPARTMENT OF
TRANSPORTATION
JUL 24 1974
LIBRARY

17. Key Words
bridges, stress history, traffic simulation, stress range, and fatigue

18. Distribution Statement
No restrictions. This document is available to the public through the National Technical Information Service, Springfield, Virginia 22151.

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 212	22. Price

Form DOT F 1700.7 (8-69)

PREFACE

The computer program described in this report was developed as a part of a project which was a continuation of work performed by the Kelly Scientific Corporation on Forecasting of Heavy Loading Patterns on Highway Bridges (1). The computer dynamic stress analysis program developed in that work, BRGSTRS, is described in this report.

The cooperation of Dr. Conrad Heinz of the University of Maryland in furnishing strain gage recordings and correlated truck data on bridges in Maryland is gratefully acknowledged.

TABLE OF CONTENTS

Page

INTRODUCTION 1

OPERATING INSTRUCTIONS 5
- Data Input 6
- Bridge Load Data Input 26
- Data Output 27
- Deck Set Up 31
- Sample Case 33

PROGRAM DESCRIPTION 39
- List of Variables 39
- Program Routines 49
  - BREXEC - Main Routine 51
  - INIT 64
  - INPTBD 67
  - GRDNET 79
  - MODULI 86
  - STIFF 102
  - STIFFL 110
  - STIFFA 113
  - DMOMNT 123
  - PNTLOD 136
  - GRDLOD 141
  - LODDIS 157
  - ASEMBL 161
  - BRGDYN 168
  - BROUT 178
  - MAXEXT 182
  - INCRMT 188
  - MATINV 191
  - MATMUL 200

LIST OF FIGURES

Figure	Title	Page
1	Traffic Simulation Based System	2
2	Axle Signature Based System	4
3	Bridge Deck	7
4	Diaphram Section	7
5	Steel Girder	8
6	Concrete Girder	8
7	Simple Span	9
8	Continuous Beam	9
9	Data Form 1	14
10	Data Form 2	17
11	Data Form 3A	21
12	Data Form 3B	22
13	Data Form 4	24
14	Deck Setup	32
15	JCL	34
16	On-Line Printout	35
17	Program Interfaces	50
18	BREXEC Program Flow Chart	53
19	INIT Program Flow Chart	65
20	INPTBD Program Flow Chart	68
21	GRDNET Program Flow Chart	80
22	MODULI Program Flow Chart	87
23	STIFF Program Flow Chart	105
24	STIFFL Program Flow Chart	111
25	STIFFA Program Flow Chart	116
26	DMOMNT Program Flow Chart	125
27	PNTLOD Program Flow Chart	137
28	GROLOD Program Flow Chart	143
29	LODDIS Program Flow Chart	158

LIST OF FIGURES (Continued

Figure	Title	Page
30	ASEMBL Program Flow Chart	163
31	BRGDYN Program Flow Chart	171
32	BROUT Program Flow Chart	179
33	MAXEXT Program Flow Chart	184
34	INCRMT Program Flow Chart	189
35	MATINV Program Flow Chart	194
36	MATMUL Program Flow Chart	201

LIST OF ABBREVIATIONS AND SYMBOLS

a	Distance from the L^{th} support to the centroid, along the X axis, of the unit load discontinuity moment area
AA	The Area of the Moment Diagram
b	Distance from the $(L+1)^{th}$ support to the centroid of the unit load discontinuity moment area
C	The distance to the neutral axis
D_R	The diameter of reinforcing steel rods
Ec	The Modulus of Elasticity for Concrete
Es	The Modulus of Elasticity for Steel
EI	The Product of the Modulus of Elasticity and the Moment of Inertia
$[F(t)]$	the force vector at time t created by the live traffic load
I	The moment of inertia
G	Gravitational constant
$[K]$	The unit load stiffness matrix
m	The mass of an element (grid) of the bridge
$M_{B_{ijL}}$	The total bending moment at the i^{th} grid, j^{th} beam and L^{th} span
M_D	The discontinuity moments for unit loads
$M_{DT_{pi}}$	The discontinuity due to the load p at location i
$M_{I_{ijL}}$	The interior moment at the i^{th} grid, j^{th} beam and L^{th} span
NL	Lane Number
t	Time in seconds

LIST OF ABBREVIATIONS AND SYMBOLS (Continued)

T_1	Thickness of the bridge deck
T_2	Thickness of a concrete lateral beam (diaphragm)
V	Vehicle speed
W	Weight
XBL	Bridge Length
X	Longitudinal distance measured from beginning of bridge span
Z_U	The sectional modulus for the upper fiber
Z_B	The sectional modulus for the lower fiber
$[\delta]$	The vertical deflection vector of the grid system in the beam direction
$[\delta']$	The deflection vector in the diaphragm direction
$\ddot{\delta}$	The acceleration of a grid element
$\dot{\delta}$	The velocity of a grid element
δ_s	A static deflection
Δt	Increment of time
Δw	grid weight
Δx	grid length, in the longitudinal direction
Δy	grid width, in the lateral direction
ν_C	Poisson's ratio for concrete
ν_S	Poisson's ratio for steel
$\sigma_{(t)}$	The dynamic bending stress
σ_B	The static bending stress
τ	The lag time between a pair of axle loads

INTRODUCTION

The work performed and the results obtained under Contract FH-11-7904 to the Federal Highway Administration, Structures and Applied Mechanics Division, for a study on "Prediction of Loadings on Highway Bridges--Phase II", is described in four separate reports, FHWA-RD-73-42, 43, 44 and 45.

A primary objective of this study was to extend the traffic simulator computer program developed by the Kelly Scientific Corporation to a useable engineering tool capable of generating bridge loads. It was further the purpose of the study to develop a finite element stress analysis program, including dynamic effects due to the live load, which directly interfaced with the load generator, produce several stress histograms for various bridge and traffic configurations, and to develop and implement an alternative method based upon analytic methods rather than traffic simulation.

Report FHWA-RD-73-42 describes the work performed in the study. This includes the background which led to the work performed, a description of the work performed and problems encountered in revising BRIGLD1. The results of sensitivity testing of BRIGLD1, development of the stress program, generated histograms, description of the analytic methods investigation, and the conclusions and recommendations are also included.

Report FHWA-RD-73-43 is the Users Manual which provides utilization instructions, data preparation instructions, output and variables definitions and a description of the BRIGLD1 computer program. The description includes a narrative description section, flow charts and program listings on the main line program and each subroutine. The use of this program for stress range prediction is illustrated in Figure 1.

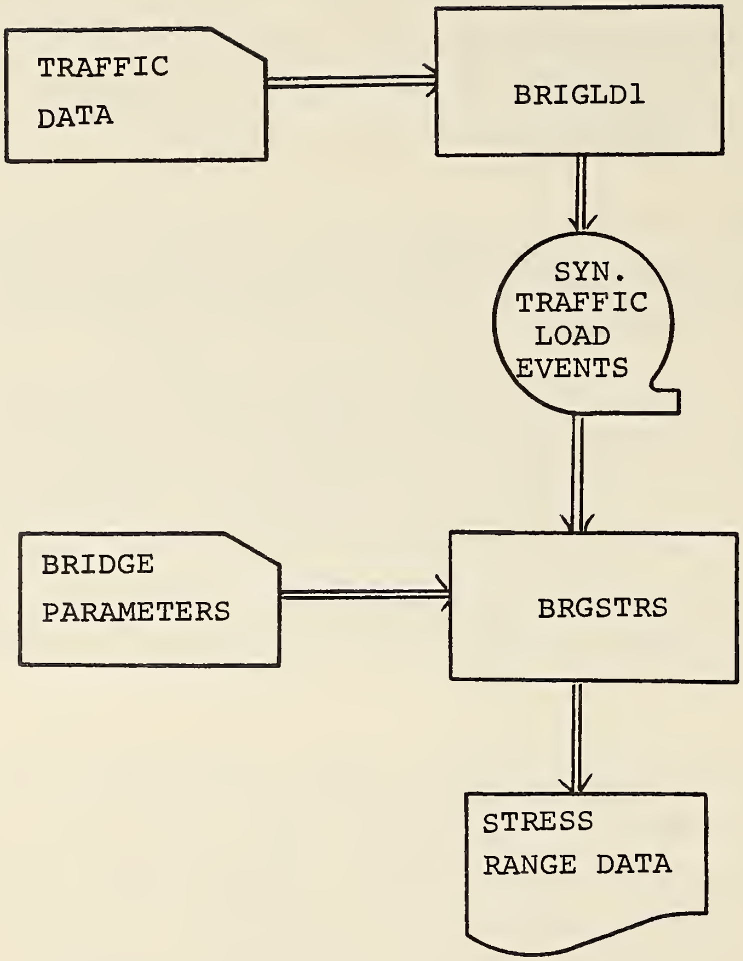


Figure 1. Traffic Simulation Based System

Report FHWA-RD-73-44 is the Users Manual for the dynamic stress analysis computer program, BRGSTRS. This report contains the same descriptive type matter as indicated above for BRIGLD1.

Report FHWA-RD-73-45 contains the Users Manuals for two computer programs which operate as a system with BRGSTRS. The first is the Synthetic Load Generator, SYNGEN, which generates single axle loads for the dynamic stress analysis program, which in turn generates a stress signature curve, trace, for each defined axle load. The second is the histogram computer program, HISGEN, which generates long-term stress range histograms from the synthetic single axle stress trace data generated by the dynamic stress analysis program, as shown in Figure 2. This is accomplished by first forming composite truck stress traces for a given truck population from the single axle data. Then, forming composite truck platoon stress traces for a given platoon population. Long-term effects are estimated from traffic density estimates and the estimated incidence of each platoon configuration. The information contained in this report, for both SYNGEN and HISGEN, is of the same form as described above for BRIGLD1.

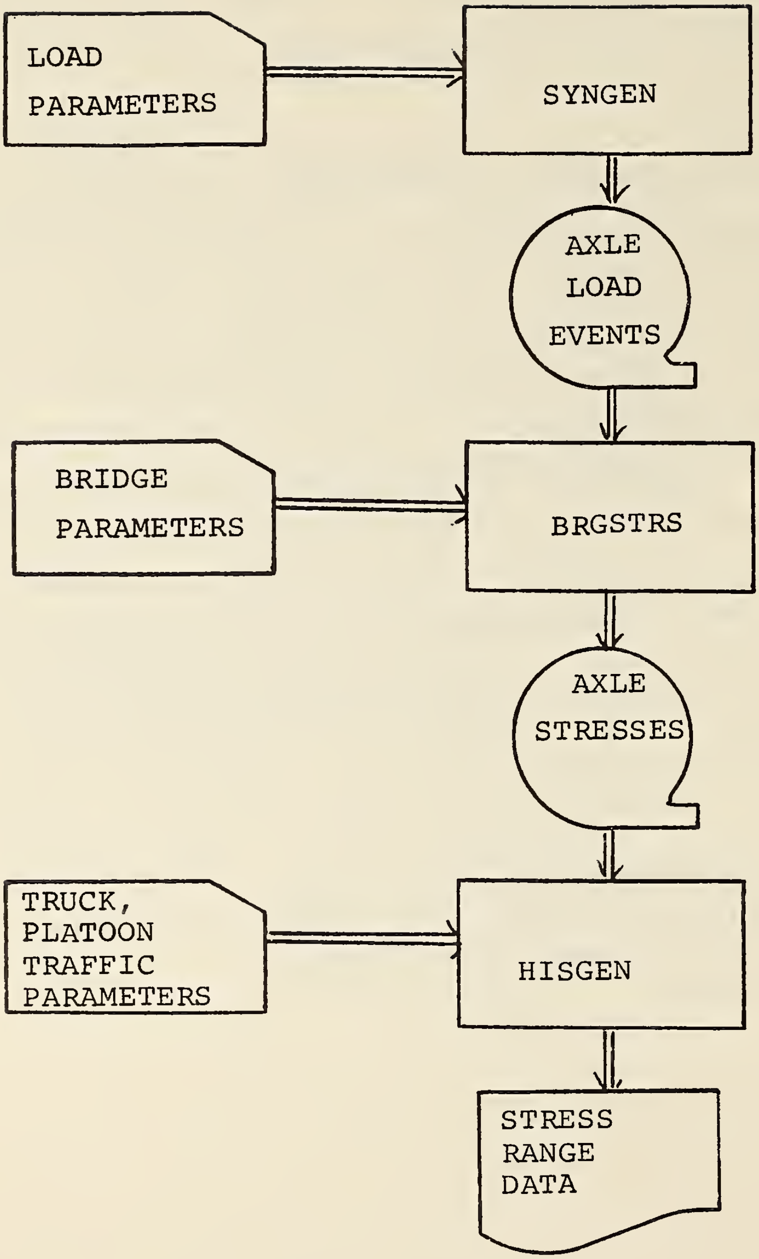


Figure 2. Axle Signature Based System

OPERATING INSTRUCTIONS

Operating instructions for the Bridge Dynamic Stress Analysis Program are divided into four sections, (1) Data Input, which includes descriptions of all data, default parameters, and data input coding forms, (2) Data Output with examples of printed output and a description of the Bridge Load Data Block, (3) Deck Set Up description, and (4) a Sample Case run.

Prior to using BRGSTRS, if a user utilizes BRIGLD1 to generate traffic data for BRGSTRS, he should verify the following:

1. That he is getting the correct distribution of trucks by type for the expected traffic.
2. That he is getting the correct density of trucks, for the simulated period of time, for the expected traffic.
3. That he is getting truck weights that correlate with the expected real truck weights.

Subsequent to verifying the above, he can then use BRGSTRS on the generated load data stored on magnetic tape.

Data Input

Data Input is divided into program header, single element data, and tabular data. Tabular data consists of Longitudinal Beam Parameters, Lateral Beam Parameters, and Output Specifications.

The program header consists of:

Card	Col	Item	Description
1	1-2	ITAPE1	Bridge Load Data input device number
1	3-4	ITAPE2	Point Stress Trace output device number
1	5-72	HEAD	68 character run header for printout and output on Point Stress Trace tape

The following figures illustrate some of the data elements to be input.

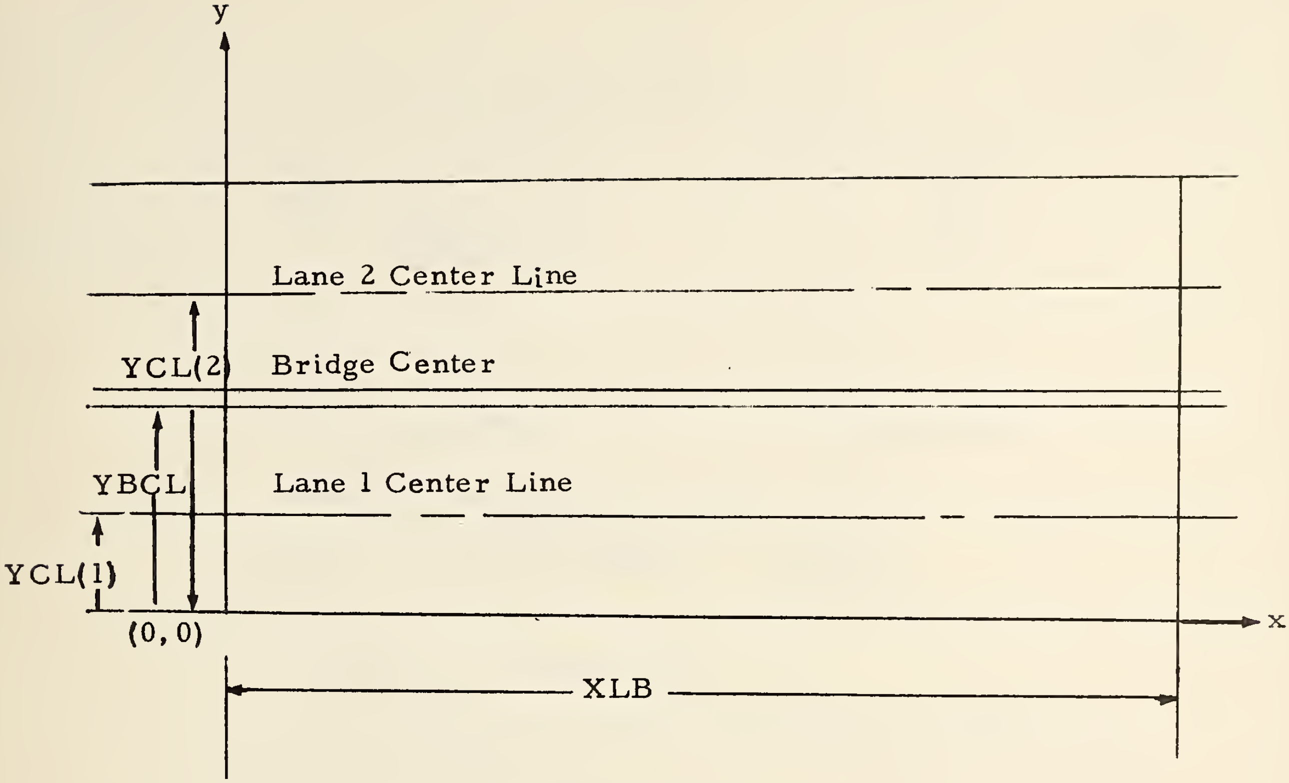


Figure 3. Bridge Deck

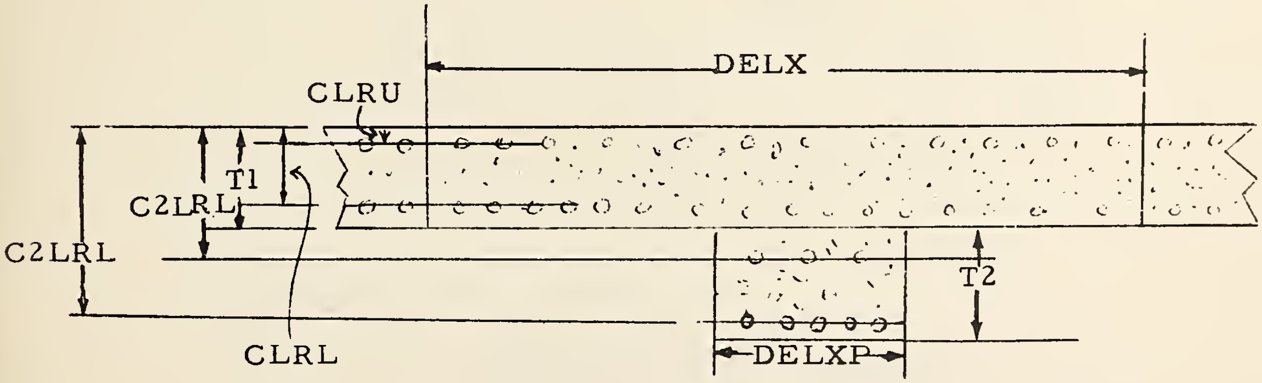


Figure 4. Diaphragm Section

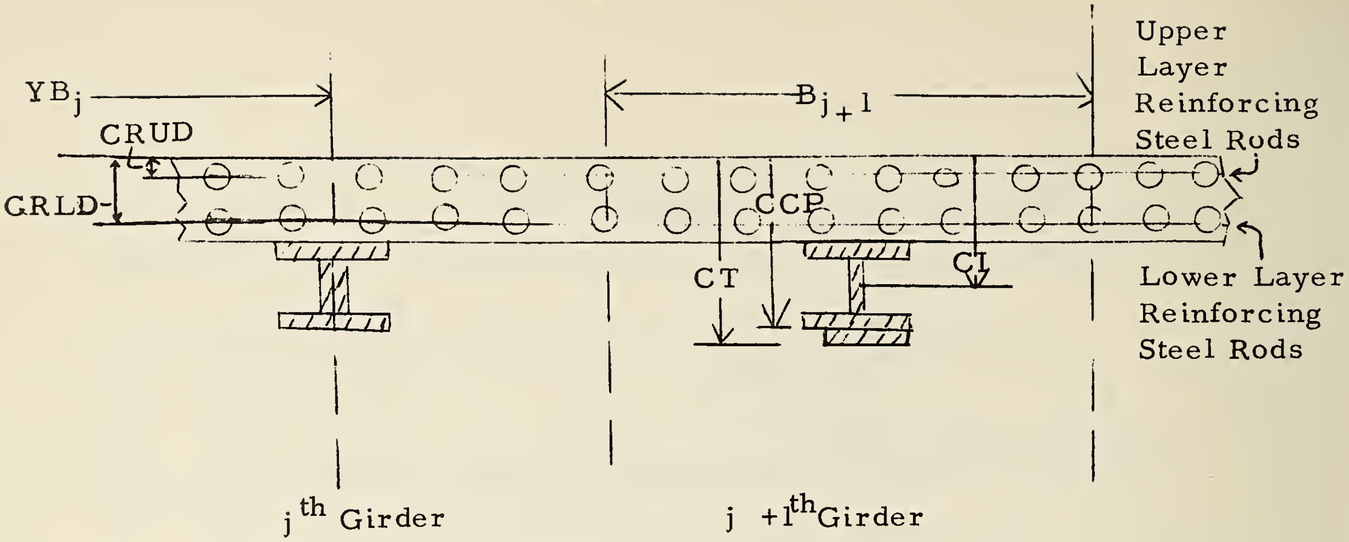

YB_j
CRUD
GRLD
B_{j+1}
Upper Layer Reinforcing Steel Rods
CCP
CT
CI
Lower Layer Reinforcing Steel Rods
j^{th} Girder
$j+1^{th}$ Girder

Figure 5. Steel Girder

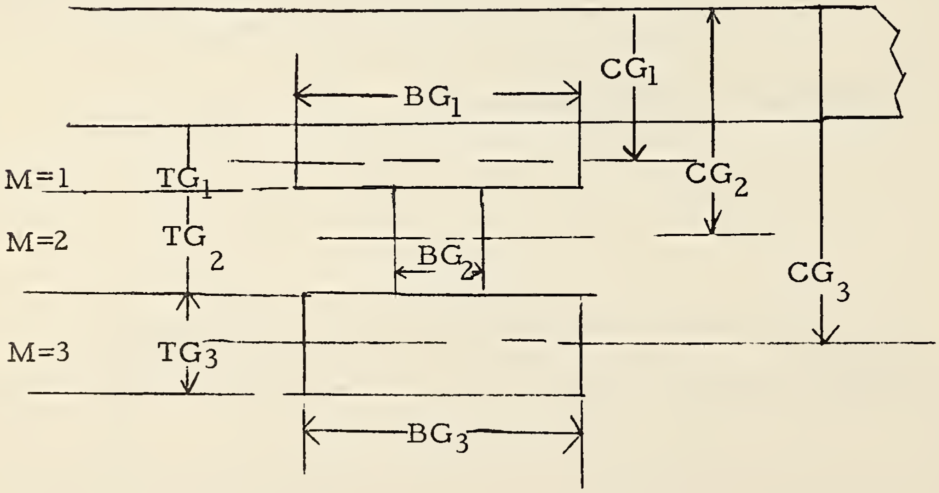


Figure 6. Concrete Girder

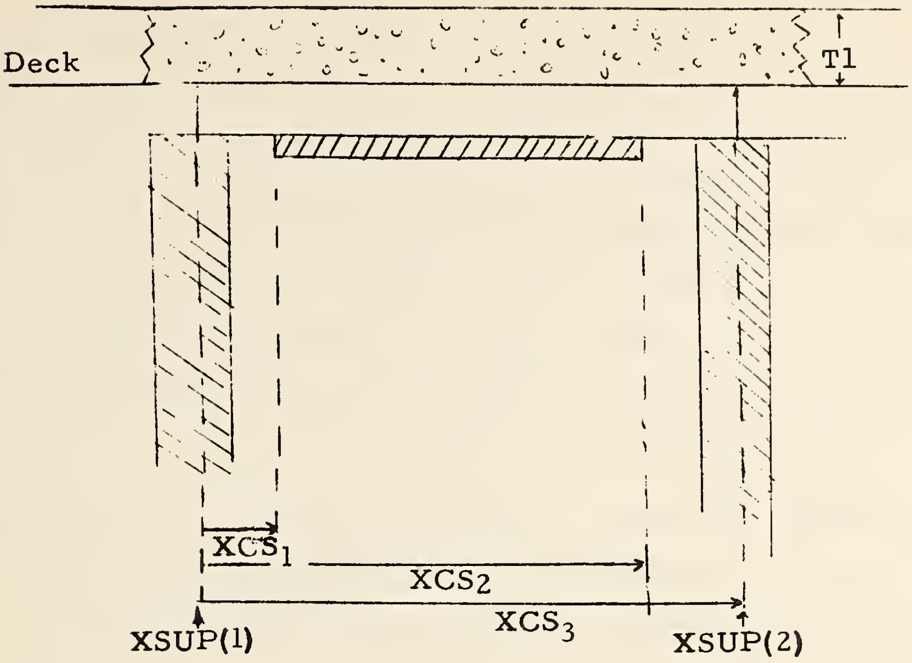


Figure 7. Simple Span

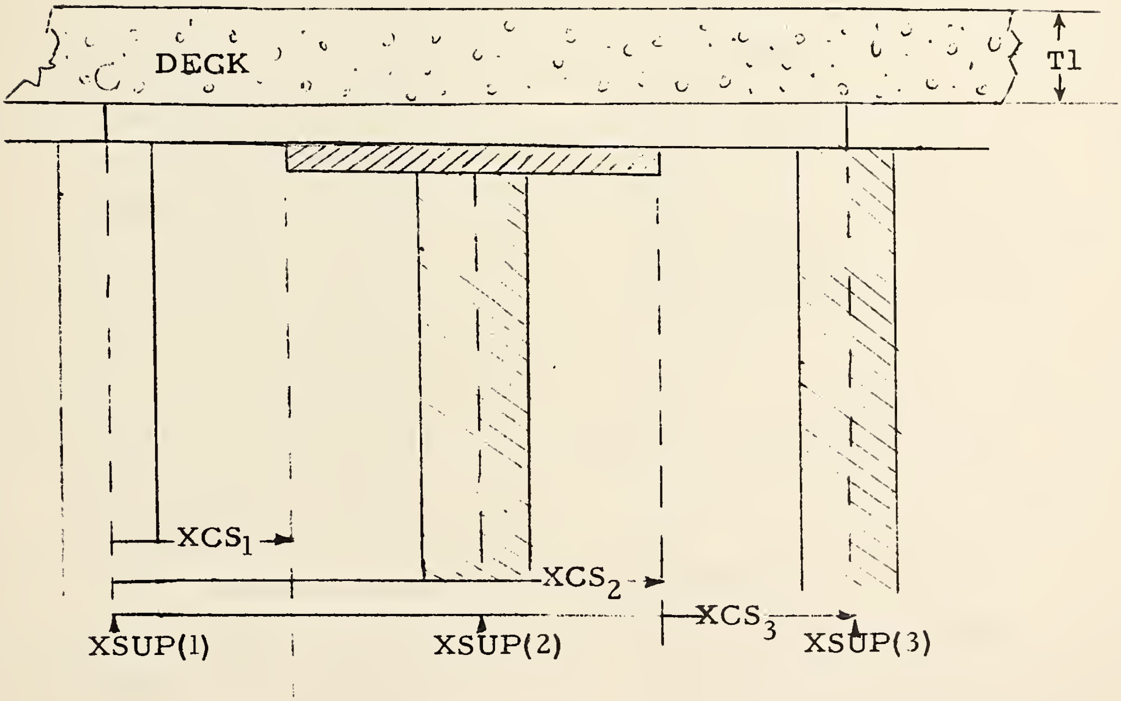


Figure 8. Continuous Beam

Single Element Data - Single element data items are input under namelist INPUTA. Only those elements with no default values need be entered and the order of entry within the namelist is arbitrary. Data Form 1 may be used for the NAMELIST input (see Figure 9). The distances CRUD and CRLD need only be entered for concrete lateral beams and the distances C2LRU and C2LRL need not be entered if effective longitudinal beam parameters are entered.

The first card $DATA and last card $END must be provided even if there is no data to be entered. A comma must follow each data element except the last. The following elements are input under namelist DATA:

ITEM	DEFAULT	UNITS	MAX.	DESCRIPTION
XLB		Ft.		Length of the bridge.
NL	2		2	Number of lanes.
YBCL		Ft.		Distance to center line of the bridge.
EC		Lb/in^2		Young's Modulus for concrete.
ES		Lb/in^2		Young's Modulus for steel.
NUC				Poisson's Ratio for concrete.
NUS				Poisson's Ratio for steel.
DT	Calculated	Sec.	DTL	Integration interval for dynamic calculations. This value is calculated if it is not input.
THRSH	300.	Psi		Stress threshold value for steady state and end of a stress event.

ITEM	DEFAULT	UNITS	MAX.	DESCRIPTION
WEIGHT		Lbs.		Weight of the bridge.
DR		In.		Diameter of the reinforcing steel rod used in the deck.
CRUD		In.		Distance from upper deck surface to center of upper layer of reinforcing steel rods.
CRLD		In.		Distance from upper deck surface to center of lower layer of reinforcing steel rods.
NB			10	Number of main longitudinal beams.
NSUP			11	Number of main beam supports.
C2LRU		In.		Distance from the top of the deck to the neutral axis of the upper layer of reinforcing steel rods, concrete diaphram.
C2LRL		In.		Distance from the top of the deck to the neutral axis of lower layer of reinforcing steel rods, concrete diaphram.
DRT		In.		Diameter of reinforcing steel rods in transverse direction.
NLRU				Number of reinforcing steel rods in the upper layer of the lateral cross section of the deck (DELX Wide).
NLRL				Number of reinforcing steel rods in the lower layer of the lateral cross section of the deck (DELX Wide).

ITEM	DEFAULT	UNITS	MAX	DESCRIPTION
CLRU		In.		Distance from the deck surface to the neutral axis of the upper layer of reinforcing steel rods in the lateral cross section of the deck.
CLRL		In.		Distance from the deck surface to the neutral axis of the lower layer of reinforcing steel rods in the lateral cross section of the deck.
ILSC				Steel or concrete diaphram indicator. 0 = steel, Not 0 = concrete
T1		In.		Mean thickness of the bridge deck.
IL				Read control: if = 0, read diaphram parameters, if not = 0, read effective diaphram parameters.
SIGTHS	300.	psi		Stress peak decay threshold.
DSIG	3000.	psi		Histogram class interval size.
NDBUG	0			Debug printout selection (10 element array). If = 0 no printout. If not = 0 print intermediate values for the following programs:

Element	Program
1	MAIN/GRDNET
2	MODULI
3	STIFF/ASEMBL
4	DMOMNT
5	PNTLOD
6	GRDLOD
7	LODDIS
8	BROUT
9	BRGDYN
10	INCRMT

CARD	COL.	ITEM	DESCRIPTION
3	1-8	XCS(1)	Distance to the coverplate, see Figure 8; ft., F8.3.
3	9-16	XCS(2)	Distance to the end of the coverplate, see Figure 8; ft., F8.3.
3	17-24	XCS(3)	Distance to the end of the beam, see Figure 8; ft., F8.3.

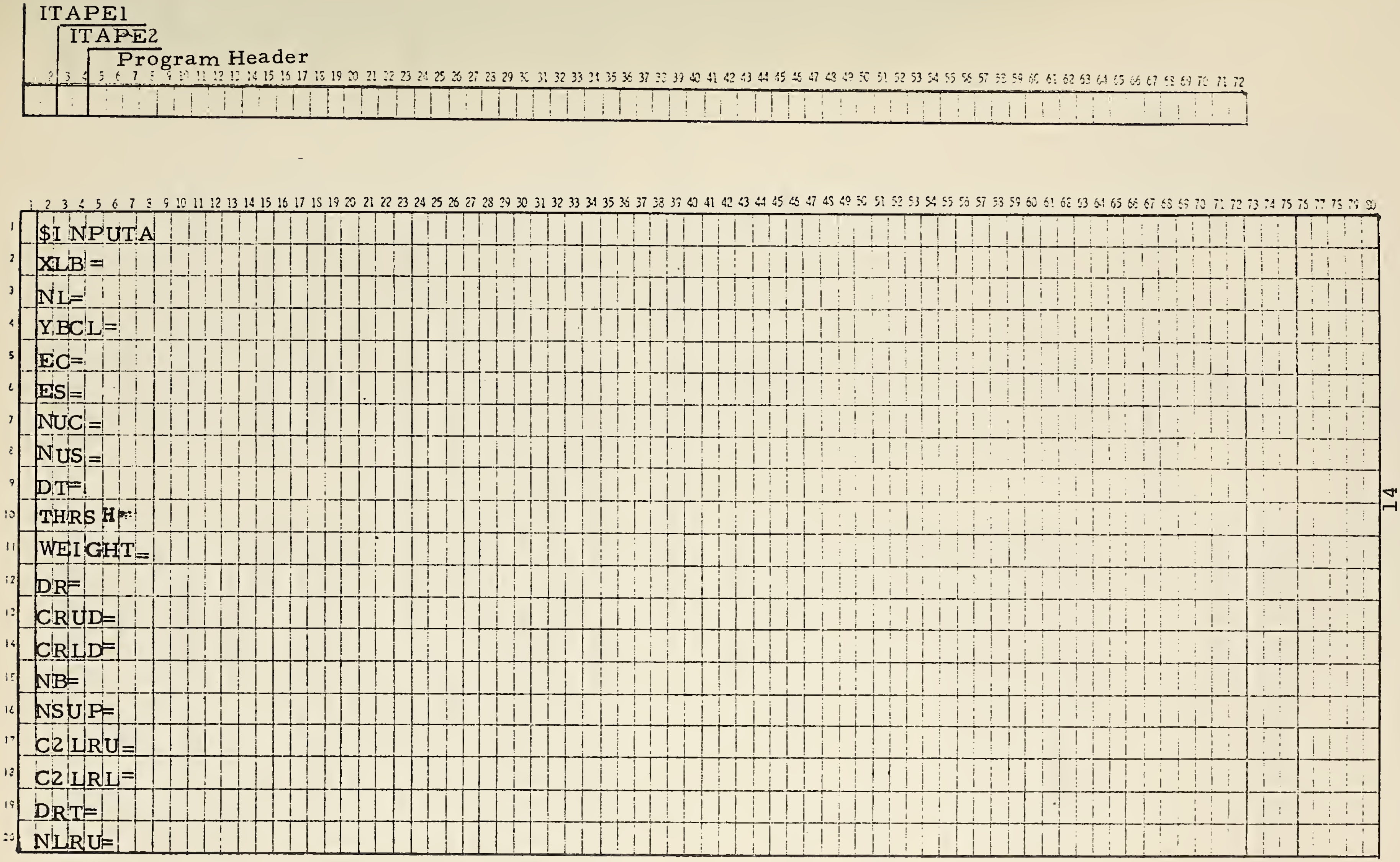
Beams may be defined in three different manners, (1) grossly by specifying the total parameters (cards 4A, Figure 10), or (2) to be calculated in detail as a steel beam (cards 4B, Figure 10), or (3) to be calculated in detail as a concrete beam (cards 4C and 4D, Figure 10). If coverplates are included, one card each of 4A, 4B or 4D must be defined for each cross section variation.

Effective Longitudinal beam parameters are:

CARD	COL.	ITEM	DESCRIPTION
4A	1-10	INRT	Effective Moment of inertia in^4 E10.3.
4A	11-20	EBAR	Effective Young's Modulus lb/in^2 E10.3.
4A	21-28	C	Effective neutral axis distance from deck surface, in., F8.3.
4A	29-38	E	Steel or Concrete Modulus depending on composition of girder lb/in^2, E10.3.

Longitudinal Steel Beam parameters are:

CARD	COL.	ITEM	DESCRIPTION
4B	1-8	AI	Girder cross-sectional area, in^2, F8.3.



ITAPE1
ITAPE2
Program Header

```
$INPUTA
XLB =
NL=
YBCL=
EC=
ES=
NUC =
NUS =
DT=
THRSH=
WEIGHT=
DR=
CRUD=
CRLD=
NB=
NSUP=
C2LRU=
C2LRL=
DRT=
NLRU=
```

Figure 9. Data Form 1

```
1 NIRL=
2 CIRU=
3 CLRL=
4 ILSC=
5 T1=
6 IL=
7 SIGTHS=
8 DSIG=
9 NDBUG= , , , , , , , , , ,
10 $END
```

YCL Lane Center Lines

```
    1           11
1A |    .    |    .    |
```

XSUP Support Positions

```
1B |  .  |  .  |  .  |  .  |  .  |  .  |  .  |  .  |  .  |  .  |  .  |
```

Figure 9. Data Form 1 (Continued)

Tabular data input follows the NAMELIST data input as follows:

Lane Center Line Positions

CARD	COL.	ITEM	DESCRIPTION
1A	1-10	YCL(1)	Center line position of Right Lane input, ft., Format F10.4.
1A	11-20	YCL(2)	Center line position of Left Lane input, ft., Format F10.4.
1B		XSUP(1)	Position of first Bridge Support (normally = 0.0) , Format 11F7.2.

Longitudinal Beam Data

The following set of data must be repeated for each effective longitudinal beam up to 10 beams. (See Figure 10.)

CARD	COL.	ITEM	DESCRIPTION
2	1-3	NRU	Number of upper reinforcing steel rods in the concrete deck - I3.
2	4-6	NRL	Number of lower reinforcing steel rods in the concrete deck - I3.
2	7-16	IUR	Moment of inertia of a steel rod in the upper part of the beam in^4, E10.3.
2	17-26	ILR	Moment of inertia of a steel rod in the lower part of the beam in^4, E10.3.
2	27-34	YB	Distance to center of beam, ft., F8.3.
2	35-42	B	Effective beam width, ft., F8.3.
2	43	NCS	Number of variations in the girder cross section, Max. = 3 (w/wo coverplates).
2	44	ISC	Steel or concrete girder index
2	45	IB	Beam data index, 0 = Read all beam parameters, not 0 = Read effective beam parameters.

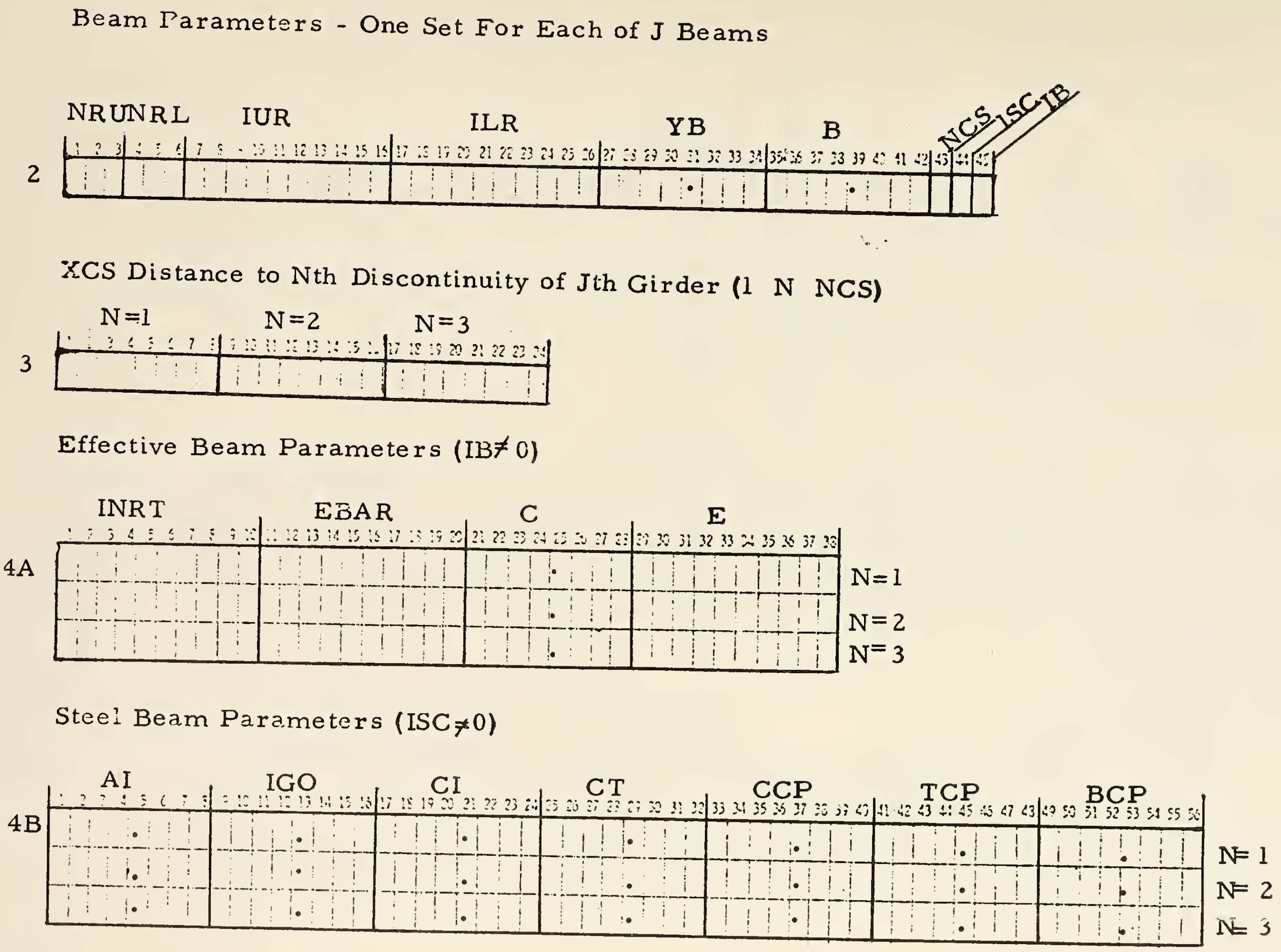

Beam Parameters - One Set For Each of J Beams
NRU
NRL
IUR
ILR
YB
B
NCS
ISC
IB
2
XCS Distance to Nth Discontinuity of Jth Girder (1 N NCS)
N=1
N=2
N=3
3
Effective Beam Parameters (IB≠0)
INRT
EBAR
C
E
4A
N=1
N=2
N=3
Steel Beam Parameters (ISC≠0)
AI
IGO
CI
CT
CCP
TCP
BCP
4B
N=1
N=2
N=3

Figure 10. Data Form 2

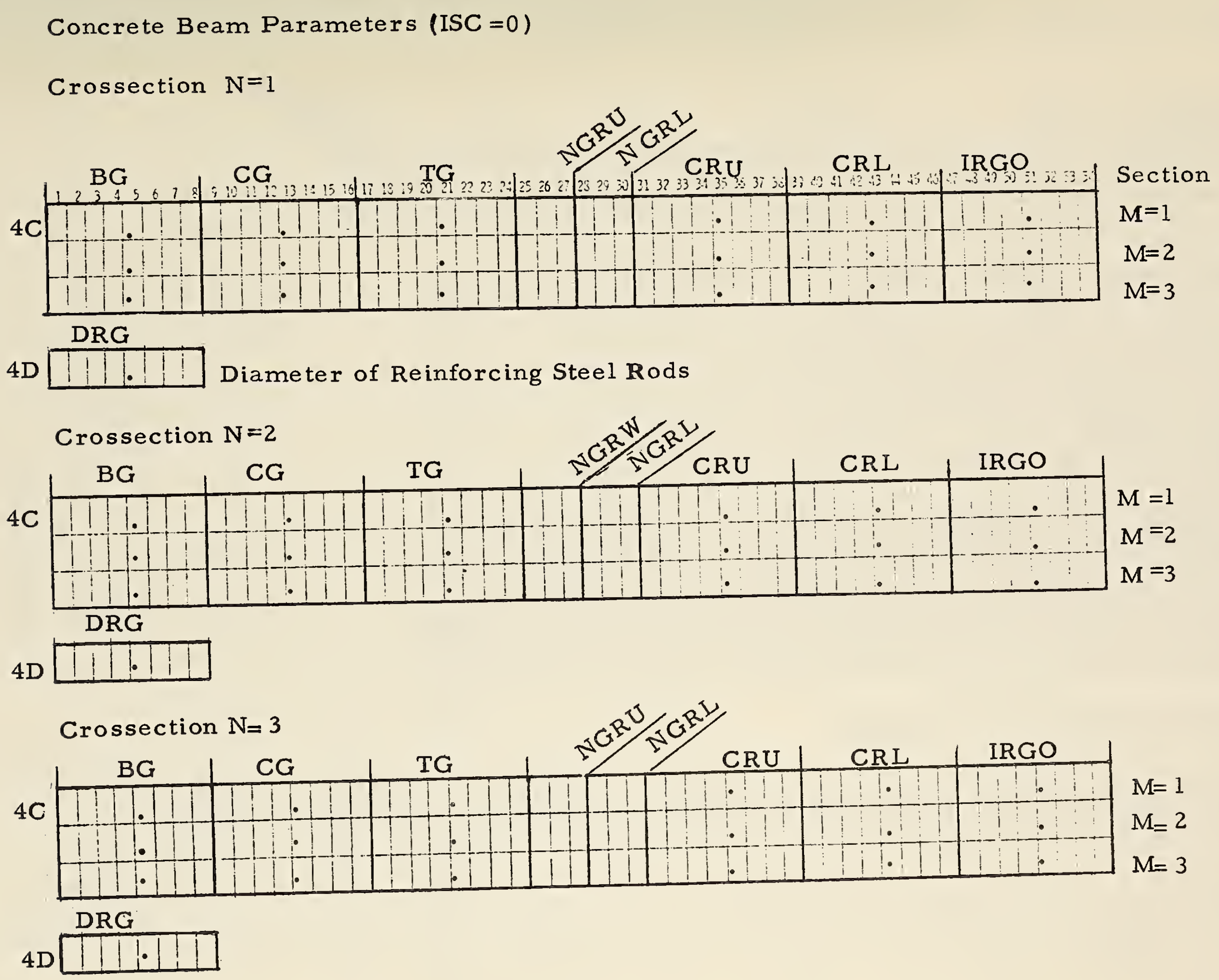


Figure 10. Data Form 2 (Cont.)

4B	9-16	IGO	Moment of inertia of the nth cross-section of the Jth girder, in^4 , F8.3
4B	17-24	CI	Distance to the neutral axis of the Nth cross section of the Jth girder, in., F8.3
4B	25-32	CT	Total distance from deck surface to the bottom of the Nth cross section, girder bottom, in., F8.3
4B	33-40	CCP	Distance to the neutral axis of the bottom cover plate of the Nth cross section, in., F8.3
4B	41-48	TCP	Thickness of plate in the Nth cross section if cover plate exists, in., F8.3

Longitudinal Concrete Beam Parameters are:

CARD	COL.	ITEM	DESCRIPTION
4C	1-8	BG	Width of the Mth section of the Nth cross section of the girder, in., F8.3
4C	9-16	CG	Distance to the neutral axis of the Mth section of the Nth cross section of the girder, in., F8.3
4C	17-24	TG	Thickness of the Mth section of the Nth cross section of the girder, in., F8.3
4C	25-27	NGRU	Number of the reinforcing steel rods in the upper layer of the Mth section of the Nth cross section of the girder, I3
4C	28-30	NGRL	Number of reinforcing steel rods in the lower layer of the Mth section of the Nth cross section of the girder, I3

4C	31-38	CRU	Distance to the neutral axis of the upper layer of reinforcing steel rods in the Mth section of the Nth cross section of the girder, in., F8.3
4C	39-46	CRL	Distance to the neutral axis of the lower layer of reinforcing steel rods in the Mth section of the Nth cross section of the girder, in., F8.3
4C	47-54	IRGO	Total Moment of inertia of the reinforcing steel rods of the Mth section of the Nth cross section of girder, in^4, F8.3
4D	1-8	DRG	Diameter of the reinforcing steel rods of the Nth cross section of the girder, in, F8.3.

Diaphragm Parameters may be input (cards 5B) or Effective Diaphragm Parameters may be used (See Figures 11 and 12.).

Effective diaphragm Parameters are:

CARD	COL.	ITEM	DESCRIPTION
5A	1-8	DELX	Grid incremental distance per span L, ft., F8.3
5A	9-18	IT	Effective Moment of inertia per DELX width in each span L, in^4, E10.3
5A	19-28	ET	Effective Young's Modulus per DELX width in each span, lb/in^2, E10.3

Diaphragm Parameters are:

Effective Diaphragm Parameters

	DELX	IT	ET	
	1 2 3 4 5 6 7 8	9 10 11 12 13 14 15 16 17 18	19 20 21 22 23 24 25 26 27 28	29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
5A1	.			
2	.			
3	.			
4	.			
5	.			
6	.			
7	.			
8	.			
9	.			
10	.			

Figure 11. Data Form 3A

Diaphragm Parameters

	DELX	ALB	ILB	N2LRU	N2URL	DELXP	T2	
	1 2 3 4 5 6 7 8	9 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24 25 26	27 28 29	30 31 32	33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48	49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
5B	.	.				.	.	
2	.	.				.	.	
3	.	.				.	.	
4	.	.				.	.	
5	.	.				.	.	
6	.	.				.	.	
7	.	.				.	.	
8	.	.				.	.	
9	.	.				.	.	
10	.	.				.	.	

Figure 12. Data Form 3B

CARD	COL.	ITEM	DESCRIPTION
5B	1-8	DELX	Grid incremental distance per span L, See Figure 4, ft., F8.3.
5B	9-16	ALB	Area of steel diaphragm in the lateral direction in^2, F8.3.
5B	17-26	ILB	Moment of inertia of steel real diaphram, in^4, E10.3
5B	27-29	N2LRU	Number of reinforcing steel rods in the upper layer of a concrete diaphram, I3
5B	30-32	N2LRL	Number of reinforcing steel rods in the lower layer of a concrete diaphram, I3
5B	33-40	DELXP	Width of the concrete diaphram, in., F8.3
5B	4--48	T2	Thickness of the concrete diaphragm, in., F8.3

Output options are specified on cards 6 and 7 (see Figure 13, Data Form 4). One card 7 must appear for each beam whether or not points are specified. Output options are:

CARD	COL	ITEM	DESCRIPTION
6	1	IOPT1	Stress graph for specified points 1 = printout, 0 = no printout
6	2	IOPT2	Stress printout for specified points 1 = printout, 0 = no printout
6	3	IOPT3	Stress histogram for specified points 1 = printout, 0 = no printout
6	4	IOPT4	Full stress printout (not recommended for long runs because of the amount of print generated) 1 = printout, 0 = no printout

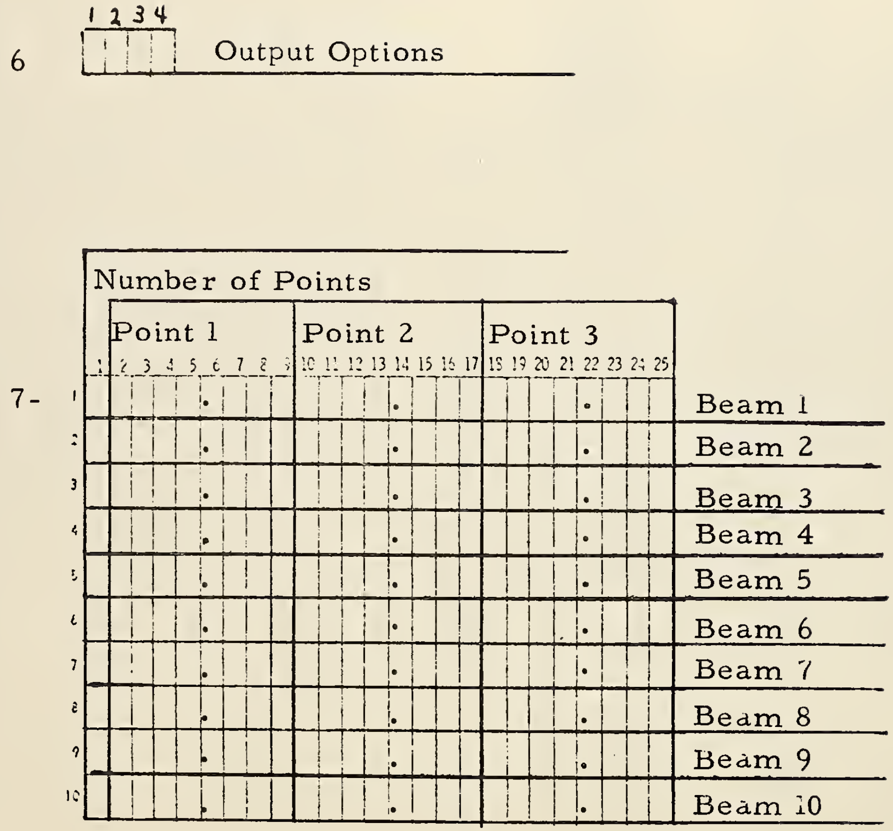


Figure 13. Data Form 4

CARD	COL	ITEM	DESCRIPTION
7	1	NPTS	Number of points to be sampled this beam, up to 3 points. If no points are desired, enter zero. I1
7	2-9	PTS	First sample point F8.3
7	10-17		Second sample point F8.3
7	18-25		Third sample point F8.3

Sample points will be converted to the nearest grid point used for calculation within the program. The converted sample points will be printed out at the beginning of the run.

A user should always specify sample points at midspan on those beams directly under the two right hand lanes. Further, when specifying the center lines of the two allowed lanes, on a bridge of more than two lanes, these center lines should be the actual center lines of the two right most lanes, i.e., where the heaviest traffic density will occur. If sample points are specified only at locations and beams where light loading will occur its possible that no output will be generated because of the stress threshold value for steady state and for the stress range threshold value for determining a point of maxima. A user can essentially out smart himself and obtain no output. There will be nothing wrong with the program, only the user's input.

Bridge Load Data Input

Bridge load data is input to the program in data blocks corresponding to a generation integration interval DTL and Event number IEVNTX. The Bridge Load Data block consists of the following elements:

Item	Dimension	Description
SUMHRX	1	Simulation time
DTL	1	Integration interval
IEVNTX	1	Event number
NOAXLX	1	Number of axle loads on-going on the bridge
NTRUKX	1	Number of trucks this event
LAST	1	Logical = True if this is the last block of this event
LTYPE	20	Truck type
WGTX	20	Weight of truck
SPDX	20	Speed of truck
LLANE	20	Lane of truck
TIMEX	20	Time truck went on the bridge
XPOSX	50	Axle position
LNUMX	50	Axle lane
WEITX	50	Axle weight
DXPOSX	50	Distance axle will travel during integration interval
ACCLRX	50	Acceleration of axle

The truck identification data, LTYPE, WGTX, SPDX. LLANE, and TIMEX, was captured as a truck went on the bridge and does not change during an event except that new trucks are added to the list as they occured during the event. The axle load data, XPOSX, LNUMX, WEITX, DXPOSX, AND ACCLRX are current values for each integration step.

Data Output

Output from the Dynamic Stress Analysis Program consists of:

- Printout of Run Input Parameters
- Optional printout:
  1. Stress graph for specified sample points.
  2. Stress printout for specified sample points.
  3. Stress histogram for specified sample points.
  4. Full stress printout.
- Point Stress Trace output tape - if device number is specified
- Debug output

Printout of Run Input Parameters - Run Parameter printout depicted belonging under sample case consists of the NAMELIST DATA printout, bridge deck specifications, longitudinal beam data, lateral beam data, output options, and sample points on each beam as input and as converted to the nearest computational grid point.

The stress graphical output consists of a plot of stress vs. time for up to three specified points on each beam. All points on the same beam will appear on the same graph. One set of plots will be generated for each event. Three distinct symbols will be used for sample points, i.e., point 1 = "*", point 2 = "+", point 3 = ".".

The stress printout for specified sample points will be a tabular printout of the minimum, maximum and range stress data for each cample point by event.

The stress histogram will consist of a histogram of maximum stress points for each sample point broken into 10% class intervals.

The full stress printout will consist of a printout for each time frame as follows:

Time (t)

Span # (1)

Beam # (1)

Position: $x_1\ x_2\ x_3 . . . . x_n$

Stress: $\sigma_{B1}\ \sigma_{B2}\ \sigma_{B3} . . . . \sigma_{BN}$

Dynamic Deflection: $\delta_1\ \delta_2\ \delta_3 . . . . \delta_n$

Dynamic Load Factor: $Dlf_1\ Dlf_2\ Dlf_3 . . . . Dlf_n$

Beam # (2)

⋮

Beam # (J)

⋮

Span # (2)

Beam # (1)

⋮

Span (N)

The point stress output tape will consist of the following:

Header:

Nev, W, V, N_{WL}, Δt, B_L, N_B, $N_p(1)$, ..., $N_p(NB)$

Data Block:

$\sigma(1,1)$, ..., $\sigma(NP(1),1)$, $\sigma(1,2)$, ..., $\sigma(NP(2),2)$, ...

$\sigma(1,NB)$, ..., $\sigma(NP(NB), NB)$

Each data block is repeated for each time point in the stress calculations for event Nev: The next event header follows immediately after the last data block of the previous event. Each header and data block is a separate record.

Data elements are defined as follows:

W	Weight of axle (real)
V	Speed of axle (real)
Δt	Time increment (real)
BL	Bridge Length (real)
Nev	Event Number (integer)
N_{NL}	Lane Number (integer)
N_B	Beam Number (integer)
Np(n)	Sample points/Beam, $1 \leq n \leq 3$ (integer)
$\sigma_{(n1,n2)}$	Stress value $1 \leq n1 \leq NP$, $1 \leq n2 \leq NB$ (real)

Debug output will be generated only on user specification and will consist of all pertiment parameters used or generated by the specified subroutine. Debug output of a subroutine is specified by setting the corresponding input parameter NDBUG(n) equal to 1. Program subroutines correspond to the following NDBUG elements:

NDBUG	(1)	GRDNET
	(2)	MODULI
	(3)	STIFF
	(4)	DMOMNT
	(5)	PNTLOD
	(6)	GRDLOD
	(7)	LODDIS
	(8)	ASEMBL
	(9)	BRGDYN
	(10)	INCRMT

Default setting of the NDBUG array is equal to zero specifying no debug printout.

Sample output may be found in the section Sample Case below.

It should be noted that the output of the input data includes:

1. An initial printout of the default data contained in the program, and

2. A subsequent printout of the new data input by a user. This data overlays the default data. Consequently, there is sometimes a residue of default data printed beyond the user specified input data. This occurs when the default data contained in any particular table exceeds that which is specified by a user. This excess data in no way affects the operations of the computer program.

Deck Set Up

The program deck should be set up as follows:

Users job card

JCL specifying program source and/or program source deck or object deck or combination of both

Data output specification - JCL consisting of a card specifying printer output on device IO if other than device 03 , a card specifying input (bridge loading), on device ITAPE1 and a card specifying output on device number ITAPE2.

Data input specification - JCL consisting of a card specifying data deck input IN.

Data Deck set up as described in section on data input above

Figure 14, Deck Set Up, further illustrates the above. An example is found below under the section Sample Case.

As a result of limited testing, BRGSTRS appears to require, for simple span bridges, from 1/6 to approximately 1/30 of the real time, simulated in the truck load data generation by BRIGLD1, for computer time.

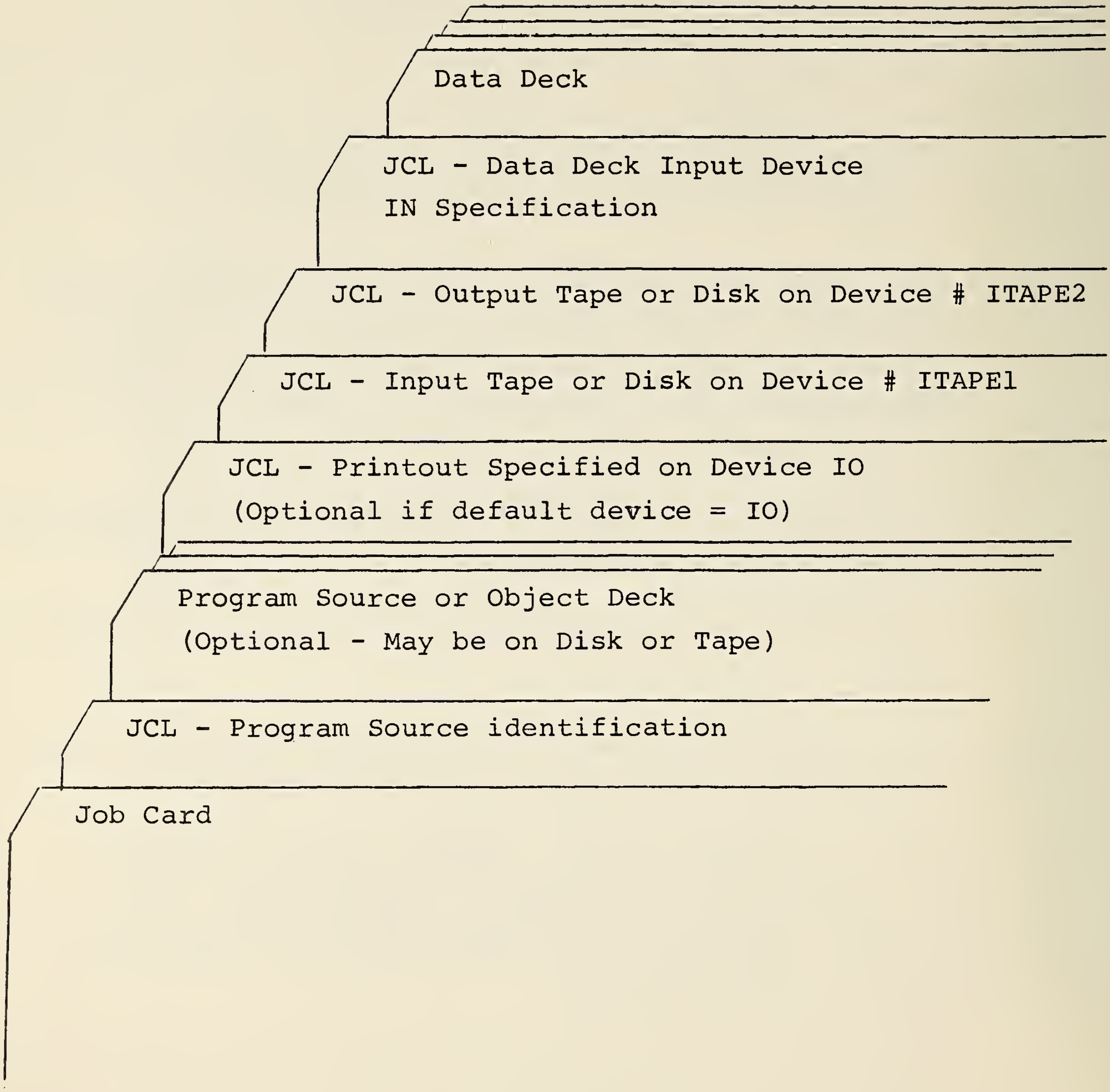

Data Deck
JCL - Data Deck Input Device
IN Specification
JCL - Output Tape or Disk on Device # ITAPE2
JCL - Input Tape or Disk on Device # ITAPE1
JCL - Printout Specified on Device IO
(Optional if default device = IO)
Program Source or Object Deck
(Optional - May be on Disk or Tape)
JCL - Program Source identification
Job Card

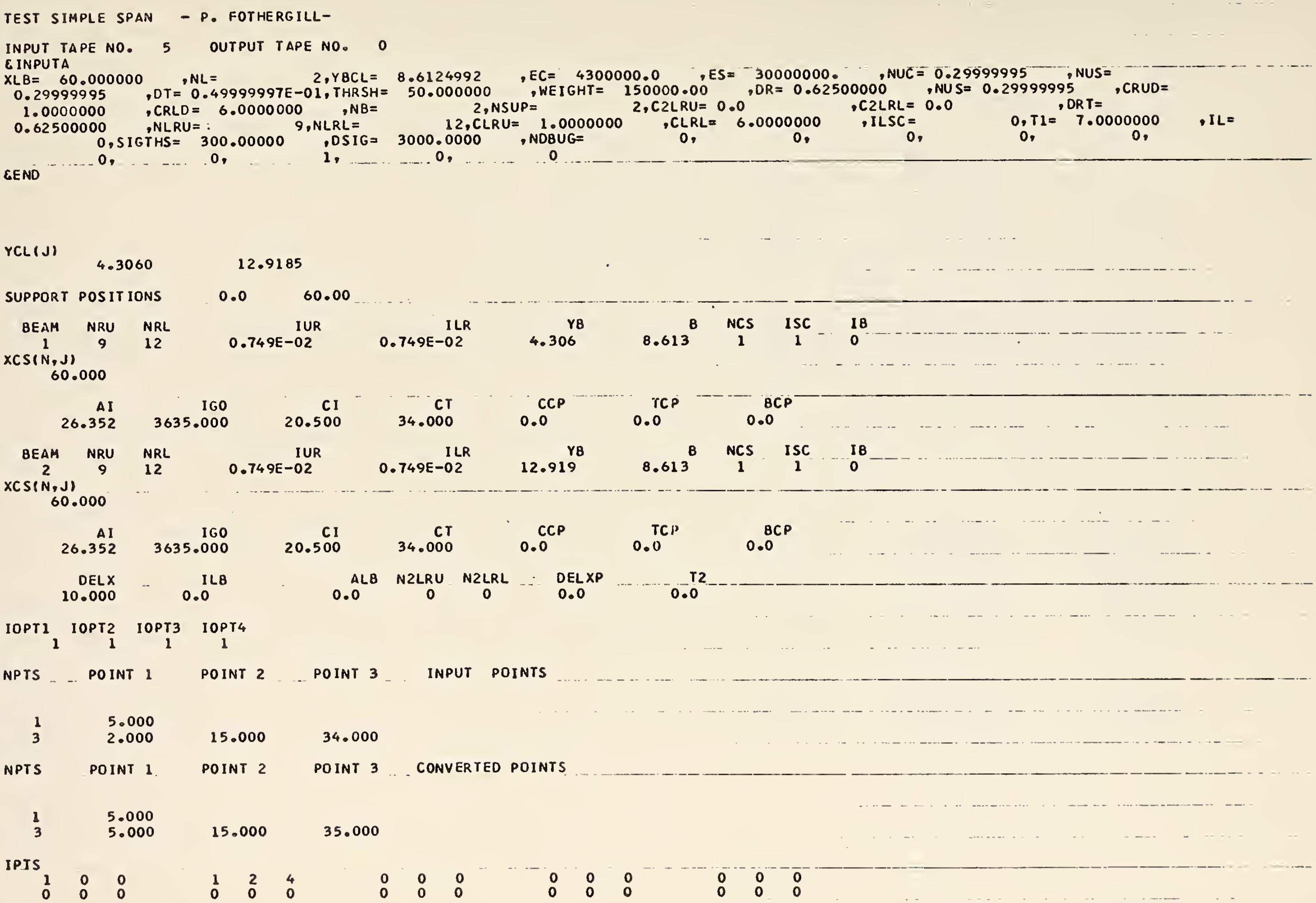
Figure 14. Deck Set Up

Sample Case

A sample case is provided here as an example for the user. It is not recommended that the source be recompiled for each run. An object program should be stored on disk if frequent use is to be made of the program. An "off-line" listing is provided of the cards, JCL, and data, used to run the program (see Figure 15). An "on-line" program printout is provided as Figure 16.

```
//R4111PF5 JOB (0118,FHRS),'PAT FOTHERGILL',TIME=20,CLASS=F,REGION=250K
/*SETUP        R4111PF5,REMOTE8
/*MESSAGE      INPUT, 007987                                         ***

/*MESSAGE      OUTPUT, 007989                                        ***

//CL EXEC FORTGCLG,PARM.FORT=(SOURCE,MAP),TIME=20
//FORT.SYSIN DD *








//LKED.SYSIN DD *








//GO.SYSUDUMP DD DUMMY,DISP=(NEW,DELETE)
//GO.FT06F001 DD SYSOUT=A,DCB=(RECFM=UA,BLKSIZE=133)
//GO.FT07F001 DD DSN=FOTHGL,UNIT=2400-4,VOL=SER=007987,
//       DISP=(OLD,KEEP),LABEL=(1,SL),
//      DCB=(RECFM=VBS,LRECL=1692,BLKSIZE=6772)


//GO.FT08F001 DD DSN=FOTHGL,UNIT=2400-4,VOL=SER=007989,
//       DISP=(NEW,KEEP),LABEL=(1,SL),
//      DCB=(RECFM=VBS,LRECL=1692,BLKSIZE=6772)
//GO.FT05F001 DD *





/*
//
```

Figure 15. JCL



```
TEST SIMPLE SPAN   - P. FOTHERGILL-

INPUT TAPE NO.   5    OUTPUT TAPE NO.   0
&INPUTA
XLB=  60.000000    ,NL=              2,YBCL=  8.6124992     ,EC=  4300000.0    ,ES=  30000000.    ,NUC= 0.29999995    ,NUS=
 0.29999995    ,DT= 0.49999997E-01,THRSH=  50.000000    ,WEIGHT=  150000.00    ,DR= 0.62500000    ,NUS= 0.29999995    ,CRUD=
  1.0000000    ,CRLD=  6.0000000    ,NB=              2,NSUP=             2,C2LRU= 0.0               ,C2LRL= 0.0              ,DRT=
 0.62500000    ,NLRU=             9,NLRL=            12,CLRU=  1.0000000    ,CLRL=  6.0000000    ,ILSC=              0,T1=  7.0000000    ,IL=
            0,SIGTHS=  300.00000    ,DSIG=  3000.0000    ,NDBUG=             0,           0,           0,           0,           0,
            0,          0,          1,          0,          0
&END


YCL(J)
        4.3060        12.9185

SUPPORT POSITIONS      0.0       60.00

  BEAM   NRU   NRL          IUR            ILR          YB          B   NCS   ISC    IB
   1      9    12        0.749E-02      0.749E-02      4.306       8.613    1     1     0
XCS(N,J)
     60.000

        AI       IGO        CI          CT          CCP         TCP         BCP
     26.352    3635.000    20.500      34.000       0.0         0.0         0.0

  BEAM   NRU   NRL          IUR            ILR          YB          B   NCS   ISC    IB
   2      9    12        0.749E-02      0.749E-02     12.919       8.613    1     1     0
XCS(N,J)
     60.000

        AI       IGO        CI          CT          CCP         TCP         BCP
     26.352    3635.000    20.500      34.000       0.0         0.0         0.0

       DELX        ILB             ALB   N2LRU   N2LRL    DELXP       T2
     10.000       0.0              0.0     0       0        0.0        0.0

IOPT1  IOPT2  IOPT3  IOPT4
   1      1      1      1

NPTS     POINT 1     POINT 2     POINT 3    INPUT  POINTS


   1       5.000
   3       2.000      15.000      34.000

NPTS     POINT 1     POINT 2     POINT 3   CONVERTED POINTS


   1       5.000
   3       5.000      15.000      35.000

IPTS
   1   0   0      1   2   4      0   0   0      0   0   0      0   0   0
   0   0   0      0   0   0      0   0   0      0   0   0      0   0   0
```

Figure 16. On-Line Printout

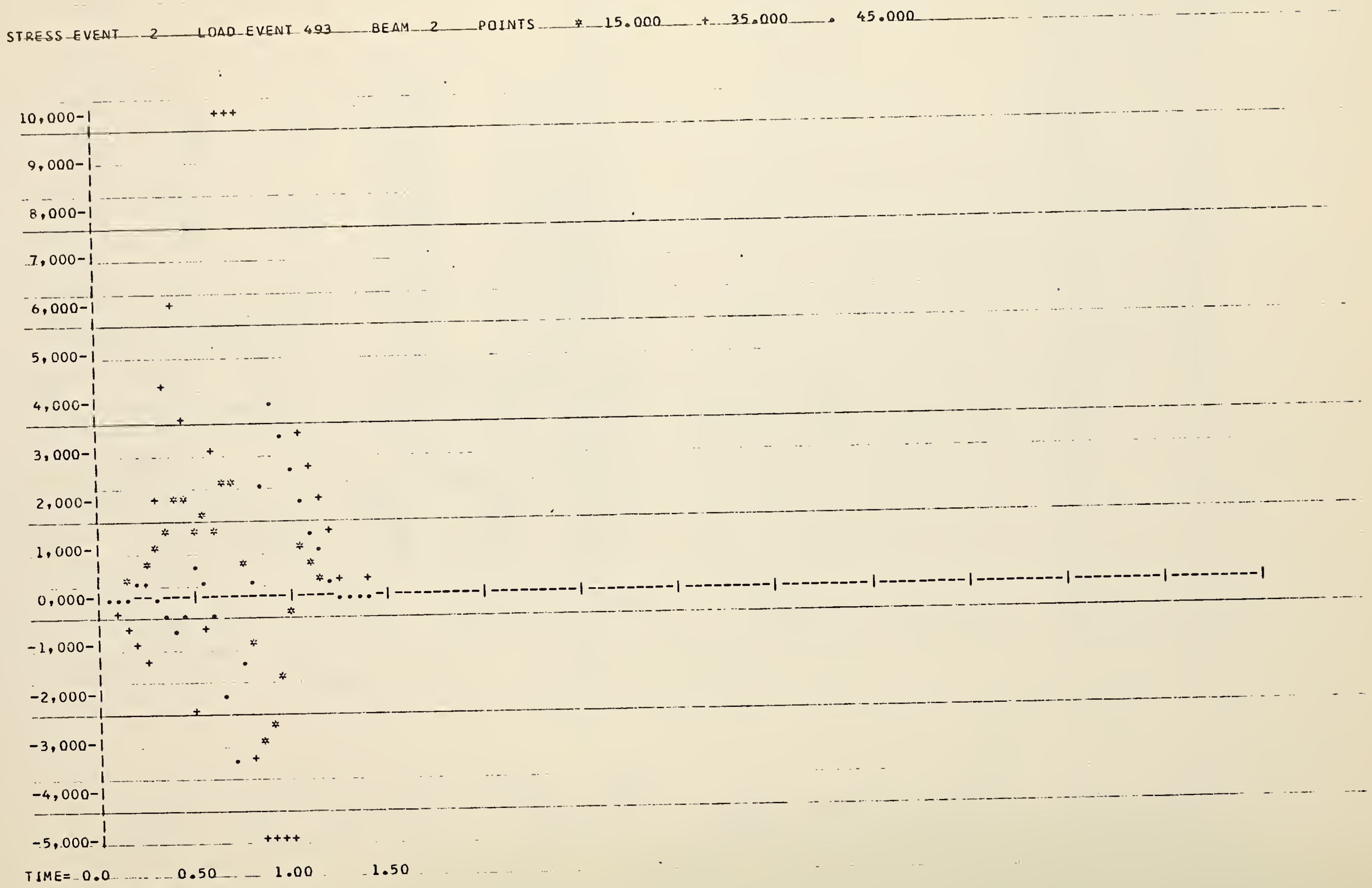


Figure 13. On-Line Printout (Continued)

```
STRESS EVENT    2    LOAD EVENT 493


BEAM NO.    2     POINT     15.000

STRESSES: MINIMUM    TIME            MAXIMUM    TIME                RANGE
               1.33  0.050           2132.35   0.400              2131.02
            1184.04  0.600           2230.21   0.650              1046.17
           -2841.16  0.850            885.78   1.050              3726.94

BEAM NO.    2     POINT     35.000

STRESSES: MINIMUM    TIME            MAXIMUM    TIME                RANGE
           -1290.49  0.250           5893.46   0.400              7183.96
           -2452.19  0.500          30310.67   0.700             32762.86
          -29919.02  0.900           3327.03   1.050             33246.05

BEAM NO.    2     POINT     45.000

STRESSES: MINIMUM    TIME            MAXIMUM    TIME                RANGE
            -512.86  0.400            577.31   0.500              1090.17
           -3349.30  0.700           4129.27   0.900              7478.57
```

Figure 16. On-Line Printout (Continued)

```
STRESS HISTOGRAMS

10% INTERVALS TO  3000.0 TO  6000.0 TO  9000.0 TO 12000.0 TO 15000.0 TO 18000.0 TO 21000.0 TO 24000.0 TO 27000.0 TO 30000.0

BEAM POINT
  2    15.000          7          0          0          0          0          0          0          0          0          0
  2    35.000          5          2          0          0          1          0          1          0          0          1
  2    45.000          6          1          0          0          0          0          0          0          0          0
```

Figure 15. On-Line Printout (Continued)

PROGRAM DESCRIPTION

Introduction

The Dynamic Stress Analysis Program is described here in terms of its individual subroutines and data interfaces. The section on each subroutine will contain a description of the routine, the program flow chart, and a program listing. Preceeding the subroutine descriptions is a list of program variables. This list contains all variables that are used by more than one subroutine or are used for program data input. Units or data type associated with the variable are provided where applicable, table dimensions, and a list of subroutines using each variable are included.

List of Variables

The program List of Variables contains a list of all program names and an identifying letter, a list of common indices and their size and meanings, and a list of all data elements which provide interfaces between program subroutines or those data elements which are required for data input or output. Consequently, program data interfaces will not be restated in the Program Routines section.

List of Variables

```
C
C
C   PROGRAM SYMBOLS    MEANING
C
C        A    ASSMBL   CALCULATE INTERNAL MOMENTS, BENDING MOMENTS,
C                          DEFLECTIONS, STIFFNESS COEFFICIENTS AND
C                          ASSEMBLE STIFFNESS MATRICES
C
C        B    BRGDYN   CALCULATE BRIDGE DYNAMICS
C
C        D    DMOMNT   CALCULATE UNIT LOAD DISCONTINUITY MOMENTS
C
C        E    ERROR    ERROR MESSAGE PRINTOUT ROUTINE
C
C        G    GRDNET   ESTABLISH GRID NETWORK
C
C        I    INPTBD   INPUT BRIDGE DATA
C
C        J    INIT     INITIALIZE DATA BLOCKS
C
C        L    LODDIS   CALCULATE ACTUAL DISCONTINUITY MOMENTS
C
C        M    MODULI   CALCULATE MODULI AND STRUCTURAL PARAMETERS
C
C        N    INCRMT   INCRIMENT TIME AND LOAD POSITIONS
C
C        P    PNTLOD   DEVELOP POINT LOADS
C
C        R    GRDLOD   CALCULATE GRID LOADS / SPAN
C
C        S    STIFF    CALCULATE LONGITUDINAL STIFFNESS COEFFICIENTS
C
C        T    STIFFL   CALCULATE LATERAL STIFFNESS COEFFICIENTS
C
C        U    STIFFA   ASSEMBLE THE STIFFNESS MATRIX
C
C
C        Q    MAXEXT   EXTRACT STRESS PEAKS
C
C        X    BREXEC   MAIN ROUTINE
C
C
C
C
C   INDICES
C        I   100   GRID PIONT INDEX
C        II        TEMPORARY INDEX
C        J   10    BEAM INDEX
C        K   20    GRID PIONTS ( ONE SPAN )
C        L   10    SPAN INDEX
C        LL  11    SUPPORT INDEX
C        M   3     CONCRETE GIRDER SECTION INDEX
C        N   3     CROSS SECTION INDEX
C
C
```

List of Variables (Continued)

```
C
C   LABEL DIM   TYPE DESCRIPTION                                         PROGRAM
C
C   AI    N,J        GIRDER CROSS-SECTIONAL AREA                         IM
C
C   ALB   L          AREA OF STEEL  REAL OR EQUIVALENT BEAM IN THE       IM
C                            LATERAL DIRECTION
C
C
C
C
C
C
C   B     J          EFFECTIVE BEAM WIDTH (FEET)                         IM
C
C   BCP   N,J        WIDTH OF COVER PLATE IN THE N TH CROSS SECTION OF   IM
C                            THE J TH BEAM
C
C   BG    M,N,J      WIDTH OF THE M TH SECTION OF THE N TH CROSS SECTION IM
C                            OF THE J TH CONCRETE GIRDER
C
C
C
C
C   C     N,J        EFFECTIVE NEUTRAL AXIS DISTANCE FROM DECK SURFACE   IM
C
C   CCP   N,J        DISTANCE TO THE NEUTRAL AXIS OF THE BOTTOM COVER    IM
C                            PLATE OF THE N TH CROSS SECTION OF THE
C                            J TH BEAM
C
C   CG    M,N,J      DISTANCE TO THE NEUTRAL AXIS OF THE M TH SECTION    I
C                            OF THE N TH CROSS SECTION OF THE J TH
C                            CONCRETE GIRDER
C
C   CI    N,J        DISTANCE TO THE NEUTRAL AXIS OF THE N TH CROSS      IM
C                            SECTION OF THE J TH GIRDER
C
```

```
C   LABEL DIM   TYPE DESCRIPTION                                  PROGRAM
C
C   CLRL             DISTANCE FROM THE DECK SURFACE TO THE NEUTRAL     IM
C                           AXIS OF THE LOWER LAYER OF REINFORCING
C                           STEEL RODS  IN THE LATERAL CROSS SECTION
C                           OF DECK
C
C   CLRU             DISTANCE FROM THE DECK SURFACE TO THE NEUTRAL     IM
C                           AXIS OF THE UPPER LAYER OF REINFORCING
C                           STEEL RODS  IN THE LATERAL CROSS SECTION
C                           OF DECK
C
C   CLT              TOTAL DISTANCE TO BOTTOM OF LATERAL BEAM =T1+T2   IM
C
C   CRL   M,N,J      DISTANCE TO THE NEUTRAL AXIS OF THE LOWER LAYER OF IM
C                           REINFORCING STEEL RODS M TH SECTION OF THE
C                           N TH CROSS SECTION OF THE J TH GIRDER
C
C   CRLD             DISTANCE FROM UPPER DECK SURFACE, TOP TO CENTER OF IM
C                           LOWER LAYER OF REINFORCING STEEL RODS OF
C                           THE DECK
C
C   CRU   M,N,J      DISTANCE TO THE NEUTRAL AXIS OF THE UPPER LAYER OF IM
C                           REINFORCING STEEL RODS M TH SECTION OF THE
C                           N TH CROSS SECTION OF THE J TH GIRDER
C
C   CRUD             DISTANCE FROM UPPER DECK SURFACE, TOP TO CENTER OF IM
C                           UPPER LAYER OF REINFORCING STEEL RODS OF
C                           THE DECK
C
C   CT    N,J        TOTAL DISTANCE FROM DECK SURFACE, TOP TO BOTTOM OF IM
C                           THE  N TH CROSS SECTION, GIRDER BOTTOM,
C                           OR BOTTOM COVER BOTTOM, OF J TH BEAM
C
C
C   C2LRL            DISTANCE FROM THE TOP OF THE DECK TO THE NEUTRAL   IM
C                           AXIS OF THE LOWER LAYER OF REINFORCING
C                           STEEL RODS, CONCRETE DIAPHRAM
C
C   C2LRU            DISTANCE FROM THE TOP OF THE DECK TO THE NEUTRAL   IM
C                           AXIS OF THE UPPER LAYER OF REINFORCING
C                           STEEL RODS, CONCRETE DIAPHRAM
C
C
```

```
C    LABEL DIM   TYPE DESCRIPTION                                      PROGRAM
C
C    DELW   L         GRID SQUARE WEIGHT PER SPAN (LB)                  GB
C
C    DELX   L         GRID INCREMENTAL DISTANCE PER SPAN (FT)           GMRA
C                             IN LONGITUDINAL DIRECTION
C
C    DELXP L          WIDTH OF THE CONCRETE DIAPHRAM                    IM
C
C    DLF    I         DYNAMIC LOAD FACTOR                               XB
C
C    DFLND  I         DYNAMIC VELOCITIES                                XB
C
C    DFLNDDI          DYNAMIC ACCELERATIONS                             XB
C
C    DFLN   I         DYNAMIC DEFLECTIONS                               XB
C
C    DR               DIAMETER OF REINFORCING STEEL ROD USED IN DECK    IM
C
C    DRG    N,J       DIAMETER OF THE REINFORCING STEEL ROD  IN THE  N TH  IM
C                            CROSS SECTION OF THE J TH  CONCRETE GIRDER
C
C    DRT              DIAMETER OF REINFORCING STEEL RODS IN TRANSVERSE  I
C                            DIRECTION
C
C    DT               INTEGRATION INTERVAL FOR DYNAMIC CALCULATIONS     IX
C
C    DTL              INTEGRATION INTERVAL FOR LOAD GENERATOR           X
C
C    DXPOSX II        DISTANCE OF TRAVEL DURING SIMULATION INTERVAL     IX
C
C
C
C
C    E      N,J       STEEL/CONCRETE MODULUS, DEPENDING ON COMPOSITION OF  EIM
C                            J TH GIRDER
C
C    EBAR   N,J       EFFECTIVE YOUNGS MODULUS                          IM
C
C    EC               YOUNG'S MODULUS FOR CONCRETE                      IM
C
C    EEI    L,J       AVERAGED PRODUCT OF THE EFFECTIVE BEAM MODULUS    DA
C                            OF ELASTICITY AND MOMENT OF INERTIA
C
C    EI     N,J       PRODUCT OF THE J TH EFFECTIVE BEAM MOMENT OF      MSDA
C                            INERTIA AT CROSS SECTION N
C
C    EIBL   N,J       EI FOR EFFECTIVE BEAM - LONGITUDINAL              M
C
C    EICG   N,J       EC * ICG                                          M
C
C    EICP   N,J       ES * ICP                                          M
C
C    EID    N,J       EC * ID                                           M
C
C    EIG    N,J       ES * IG OR EICG + EIRG                            M
C
C    EIL    L         EI FOR LATERAL SECTION L TH SPAN                  MS
```

```
C
C   EILB   L        ES * ILB OR EILBR + EILBC                             M
C
C   EILBC  L        EC * ILBC                                             M
C
C   EILBR  L        ES * ILBR                                             M
C
C   EILD   L        EILDC + EILR                                          M
C
C   EILDC  L        EC * ILDC                                             M
C
C   EILR   L        ES * ILR                                              M
C
C   EIR    N,J      ES * IR                                               M
C
C   EIRG   N,J      ES * IRG                                              M
C
C   EPSLON          PERCENT ERROR ALLOWED IN THE NUMERICAL SOLUTION       I
C                            OF THE NON-HOMOGENOUS DIFFERENTIAL
C                            EQUATIONS
C
C   ES              YOUNGS MODULUS FOR STEEL                              IM
C
C   ET     L        EFFECTIVE YOUNGS MODULUS PER DX WIDTH IN EACH SPAN    IM
C
C
C
C   F      I,J                                                            RLA
C
C
C
C
C   IB     J        IF =0 READ ALL BEAM PARAMETERS                        I
C                            IF NOT =0 READ EFFECTIVE BEAM PARAMETERS
C
C   IBL    N,J      CALCULATED MOMENT OF INERTIA - LONGITUDINAL - OF      M
C                            EFFECTIVE BEAM WIDTH = INRT(N,J)
C
C   ICG    N,J      MOMENT OF INERTIA                                     M
C
C   ICP    N,J      MOMENT OF INERTIA - COVER PLATE                       M
C
C   ID     N,J      MOMENT OF INERTIA - DECK                              M
C
C   IE              EVENT NUMBER - BRIDGE                                 XO
C
C   IEVENT          EVENT NUMBER - LOAD GENERATOR                         WZ
```

List of Variables (Continued)

```
C    LABEL DIM   TYPE DESCRIPTION                                         PROGRAM
C
C
C    IG    N,J    R   MOMENT OF INERTIA                                   M
C    IGO   N,J    R   MOMENT OF INERTIA OF N TH CROSS SECTION OF J TH     IM
C                            GIRDER
C
C    II    N,J    R   TOTAL MOMENT OF INERTIA                             M
C
C    IL               IF =0 READ LATERAL BEAM PARAMETERS                  I
C                            IF NOT =0 READ EFFECTIVE BEAM PARAMETERS
C
C    ILB   L      R   MOMENT OF INERTIA OF STEEL REAL OR EQUIVALENT       IM
C                            BEAM
C
C    ILBC  L      R   MOMENT OF INERTIA                                   M
C
C    ILBR  L      R   MOMENT OF INERTIA                                   M
C
C    ILDC  L      R   MOMENT OF INERTIA                                   M
C
C    ILR   J      R   TOTAL MOMENT OF INERTIA OF A STEEL ROD IN THE       IM
C                            LOWER PART OF THE DIAPHRAM
C
C    ILSC             STEEL OR CONCRETE DIAPHRAM INDICATOR                IM
C                            0 = STEEL, NOT 0 = CONCRETE
C
C    IN               INPUT DEVICE NUMBER                                 I
C
C    INRT  N,J    R   EFFECTIVE MOMENT OF INERTIA FOR THE J TH EFFECTIVE  IM
C                            BEAM
C
C    IO               OUTPUT DEVICE NUMBER                                IMGPR
C
C    IR    N,J    R   MOMENT OF INERTIA                                   M
C
C    IRG   N,J    R   MOMENT OF INERTIA                                   M
C
C    IRGO  M,N,J  R   TOTAL MOMENT OF INERTIA OF THE STEEL REINFORCING    IM
C                            RODS OF THE M TH SECTION OF THE N TH CROSS
C                            SECTION OF THE J TH GIRDER
C
C    IS    L          NUMBER OF GRIDS LONGITUDINALLY ON A GIVEN SPAN      GDESRLA
C
C    ISC   J          STEEL OR CONCRETE GIRDER INDEX                      IM
C                            NOT 0 = STEEL,   0 = CONCRETE
C
C    ISS              STEADY STATE INDICATOR 0= STEADY STATE,
C                            NOT 0 = TRANSIENT STATE
```

```
C   LABEL DIM   TYPE DESCRIPTION                                      PROGRAM
C
C   IT    L       R  EFFECTIVE DIAPHRAM MOMENT OF INERTIA PER DX WIDTH  IM
C                           IN EACH L SPAN
C
C   ITIME            OUTPUT TIME POINT INDEX                            XOQ
C
C   IUR   J       R  MOMENT OF INERTIA OF A STEEL ROD IN THE UPPER      IM
C                           PART OF A LATERAL BEAM
C
C   KTIME            OUTPUT TIME POINT INDEX                            XOQ
C
C
C
C   MB    I          BENDING MOMENTS                                    AR
C
C   MD    LL,K,J R   UNIT DISCONTINUITY MOMENTS AT SUPPORTS LL, FOR     DL
C                           EACH BEAM J
C
C   MDT   LL,J    R  DISCONTINUITY MOMENTS AT SUPPORTS LL, FOR EACH     LA
C                           BEAM J
C
C   MI    I          INTERIOR MOMENTS                                   A
C
C
C   NA               NUMBER OF AXLES FROM DATA BLOCK                    PR
C
C   NAP              NUMBER OF LONGITUDINAL POINT LOADS                 PGR
C                           (USUALLY EQUAL TO NA)
C
C   NB               NUMBER OF MAIN LONGITUDINAL BEAMS                  IMASRLG
C
C   NCS   J          NUMBER OF VARIATIONS IN THE J TH MAIN GIRDER CROSS IMS
C                           SECTION
C
C   NDBUG            DEBUG PRINTOUT OPTIONS
C
C   NGRL  M,N,J      NUMBER OF REINFORCING STEEL RODS IN THE LOWER LAYER IM
C                           OF THE M TH SECTION OF THE N TH CROSS
C                           SECTION OF THE J TH CONCRETE GIRDER
C
C   NGRU  M,N,J      NUMBER OF REINFORCING STEEL RODS IN THE UPPER LAYER IM
C                           OF THE M TH SECTION OF THE N TH CROSS
C                           SECTION OF THE J TH CONCRETE GIRDER
C
C   NI               DTLOAD / DT                                        XY
C
C   NL               NUMBER OF LANES                                    I
C
C   NLRL             NUMBER OF REINFORCING STEEL RODS IN THE LOWER LAYER IM
C                           OF THE LATERAL CROSS SECTION OF DECK
C
C   NLRU             NUMBER OF REINFORCING STEEL RODS IN THE UPPER LAYER IM
C                           OF THE LATERAL CROSS SECTION OF DECK
```

List of Variables (Continued)

```
C    LABEL DIM   TYPE DESCRIPTION                                      PROGRAM
C
C    NPTS  J          NUMBER OF SAMPLE POINTS PER BEAM                 IGBO
C
C    NRL   J          NUMBER OF LOWER REINFORCING STEEL RODS IN EACH   IM
C                              BEAM SECTION
C
C    NRU   J          NUMBER OF UPPER REINFORCING STEEL RODS IN EACH   IM
C                              BEAM SECTION
C    NSI              NSUP - 1                                       IXYGMDASRL
C
C    NSUP             NUMBER OF MAIN BEAM SUPPORTS  2 <= NSUP <= 11     IXYL
C
C    NUC            R POISSON'S RATIO FOR CONCRETE                     I
C
C    NUS            R POISSON'S RATIO FOR STEEL                        I
C
C    NXPTS            NUMBER OF X GRID POINTS                          GBOD
C
C    N2LRL L          NUMBER OF REINFORCING STEEL RODS IN THE LOWER LAYER  IM
C                             OF THE REAL OR EQUIVALENT CONCRETE DIAPHRAGM
C
C    N2LRU L          NUMBER OF REINFORCING STEEL RODS IN THE UPPER LAYER  IM
C                             OF THE REAL OR EQUIVALENT CONCRETE DIAPHRAGM
C
C    PTS   JJ,J       SAMPLE POINTE FOR STRESS OUTPUT                  IGBO
C
C    SIG   I          DYNAMIC BENDING STRESSES                         XB
C
C    SIGB  I,J        STATIC BENDING STRESSES                          AB
C
C    SIGP  JJ,J,KK    DYNAMIC BENDING STRESSES AT SAMPLE POINTS        XO
C
C    SK    II,II,J                                                     SA
C
C    SKP   J,J                                                         SA
C
C     STEADY        L STEADY STATE INDICATOR                           XB
C
C
C     T               TIME
C
C    TCP   N,J        THICKNESS OF EACH PLATE IN THE N TH CROSS SECTION    IM
C                             OF THE  J TH BEAM
C
C    TG    M,N,J      THICKNESS OF THE M TH SECTION OF THE N TH CROSS      IM
C                             SECTION OF THE  J TH CONCRETE GIRDER
C
C    THRSH            THRESHOLD VALUE TO DETERMINE STEADY STATE        XIB
C
C    T1               MEAN THICKNESS OF BRIDGE DECK (FEET)             IM
C
C    T2    L          THICKNESS OF A CONCRETE DIAPHRAGM                IM
```

List of Variables (Continued)

```
C   LABEL DIM  TYPE DESCRIPTION                                          PROGRAM
C
C   WP    IP,J      WEIGHT PER LOAD POINT  1.=IT.=4                      PR
C
C   WEIGHT          WEIGHT OF THE BRIDGE (LBS)                           IG
C
C
C   X     I         GRID CENTER POSITIONS  (FT)                          GDSRA
C
C   XCS   N,J       LONGITUDINAL DISTANCE TO THE N TH DISCONTINUITY IN   ISD
C                             THE J TH GIRDER (FEET)
C
C   XIL   L         MOMENT OF INERTIA - LATERAL SECTION PER L TH SPAN    M
C
C   XILB  J,L       AVERAGED MOMENT OF INERTIA OF EFFECTIVE BEAM         A
C
C   XILR  L         MOMENT OF INERTIA OF THE RODS IN A LATERAL SECTION
C                             OF THE L TH SPAN
C
C   XLB             LENGTH OF BRIDGE (FEET)                              IGR
C
C   XP    II        POSITION OF AXLE LOADS                               ZPR
C
C   XSUP  L         DISTANCE TO THE CENTER OF THE L TH SUPPORT (FEET)    IXGSDA
C
C
C   YB    J         DISTANCE TO CENTER OF MAIN BEAM, GIRDER              IR
C
C   YBCL            DISTANCE TO CENTER LINE OF BRIDGE (FEET)             ISR
C
C   YCL   J         DISTANCE TO CENTER LINE OF THE J TH LANE(FEET)       IP
C
C
C   YP    JT        TRACK POSITION                                       PR
C
C   ZBL   N,J       CROSS SECTIONAL MODULUS - EFFECTIVE LONGITUDINAL     M
C                             BEAM - LOWER HALF
C
C   ZBU   N,J       CROSS SECTIONAL MODULUS - EFFECTIVE LONGITUDINAL     MD
C                             BEAM - UPPER HALF
C
C   ZCL   L,J       AVERAGED CROSS SECTIONAL MODULUS - LOWER             DA
C
C   ZCU   L,J       AVERAGED CROSS SECTIONAL MODULUS - UPPER             DA
C
C   ZL    N,J                                                            M
C
C
C
C
C
C
C
```

Program Routines

The following describes the Dynamic Stress Analysis Program in terms of its individual subroutines. The program functions, equations solved, program interfaces are described, followed by the program flow chart and program listing. Figure 17, Program Interfaces, depicts the hirearchy of subroutine calls.

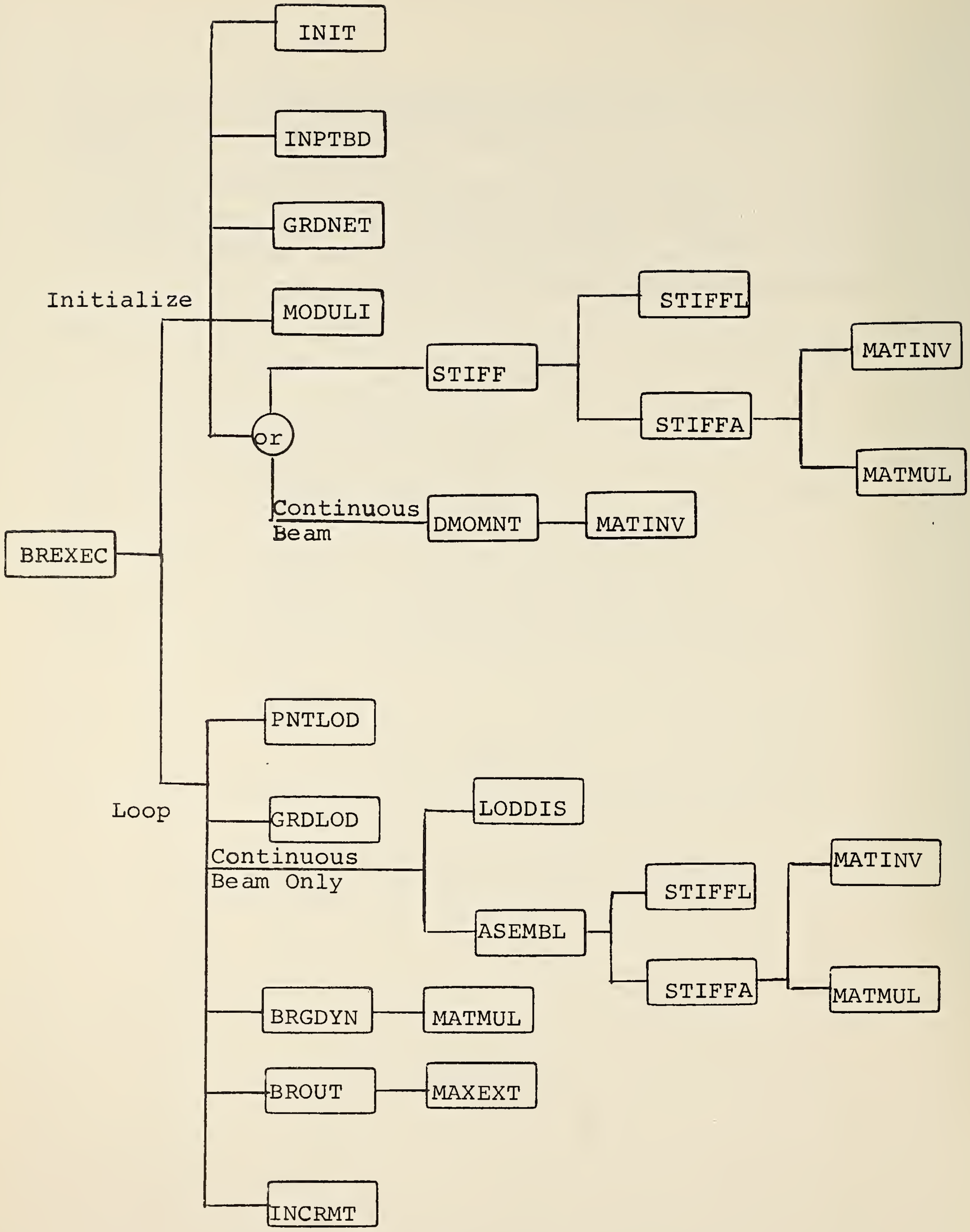


Figure 17. Program Interfaces

BREXEC

The main program, BREXEC reads in program input output device selection options and output tape header. It calls the subroutine INIT to clear data buffers and reads all input data through subroutine INPTBD. It performs all program initialization through calls to GRDNET, MODULI, STIFF, and DMOMNT. The structural event number is initialized.

The main program loop is then entered and proceeds as follows:

1. Read bridge load data block.

2. If it is the first pass, a number of intervals and an integration interval is calculated.

3. Point loads are calculated PNTLOD.

4. Loads at grid points are calculated GRDLOD.

5. If a continuous beam calculate actual load discontinuity moments-LODDIS, interior moments and deflections, bending moments and stresses and assemble stiffness coefficient - ASEMBL.

6. Calculate bridge dynamics and dynamic stress and load factor - BRGDYN.

7. If a steady state has been reached the event stresses are output - BROUT and the event number is updated.

8. Load positions are incremented INCRMT and the loop proceeds from 3 above or a new data block is read - 1 above.

The program terminates when the bridge load data is exhausted.

The following subroutines are called by this program:

- INIT
- INPTBD
- GRDNET
- MODULI
- STIFF
- DMOMNT
- PNTLOD
- GRDLOD
- BRGDYN
- BROUT
- INCRMT

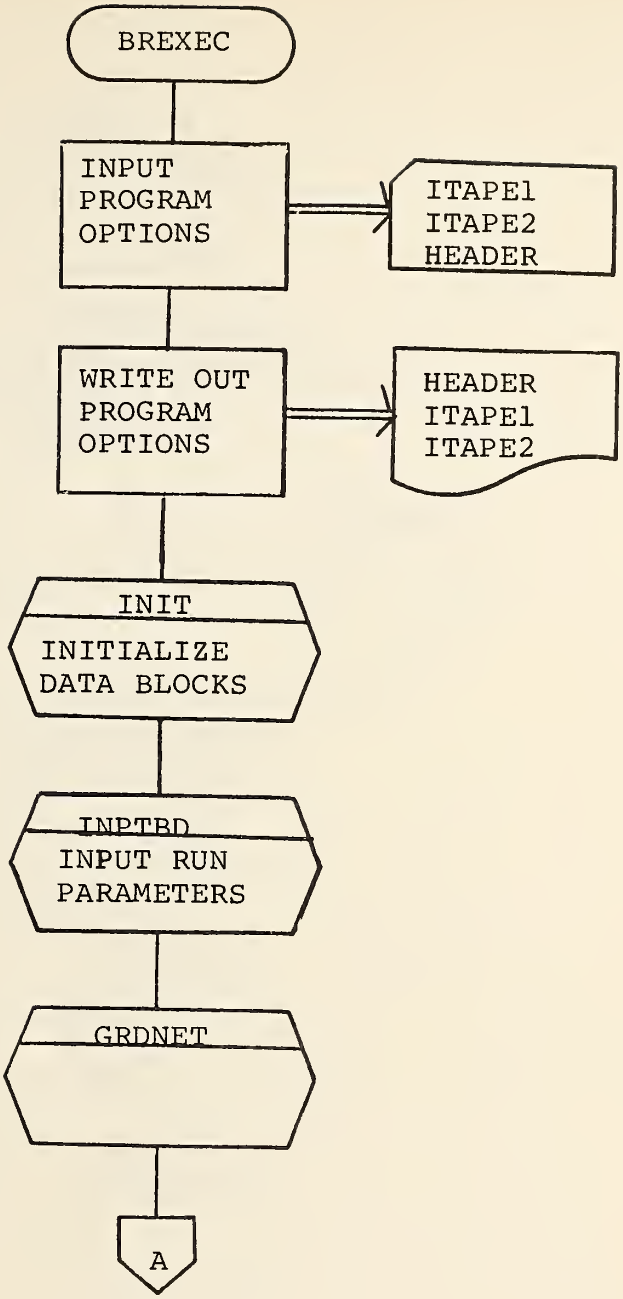


Figure 18. BREXEC Program Flow Chart

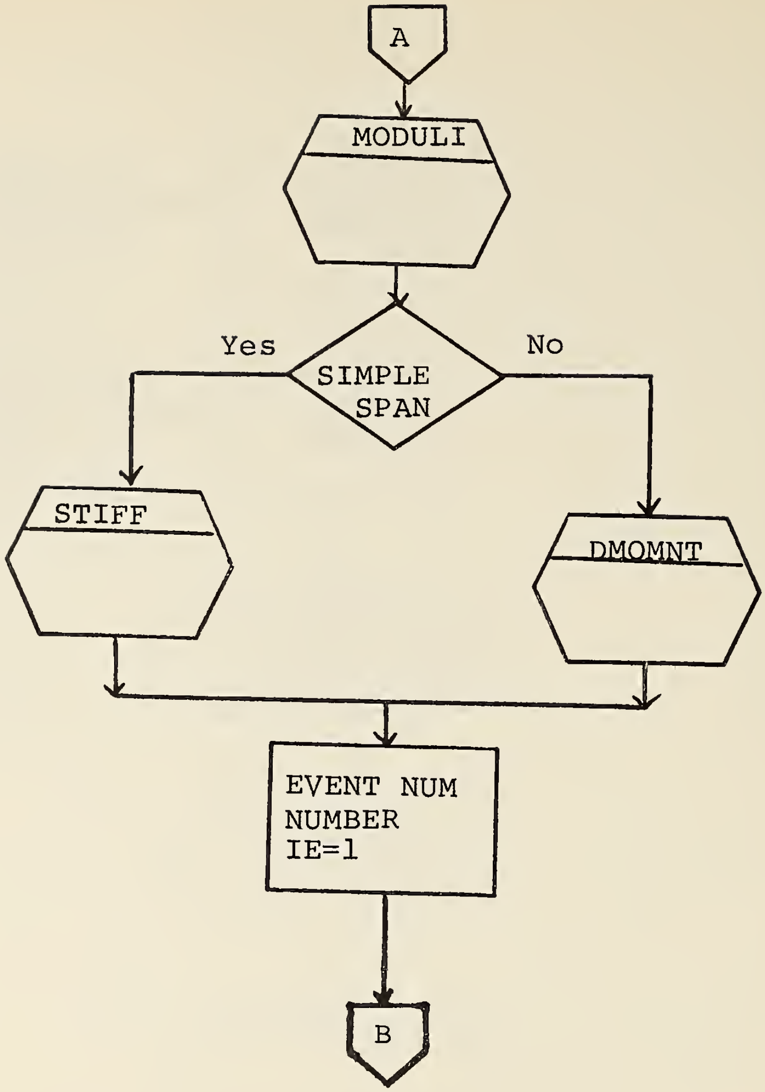


Figure 18. BREXEC Program Flow Chart (Continued)

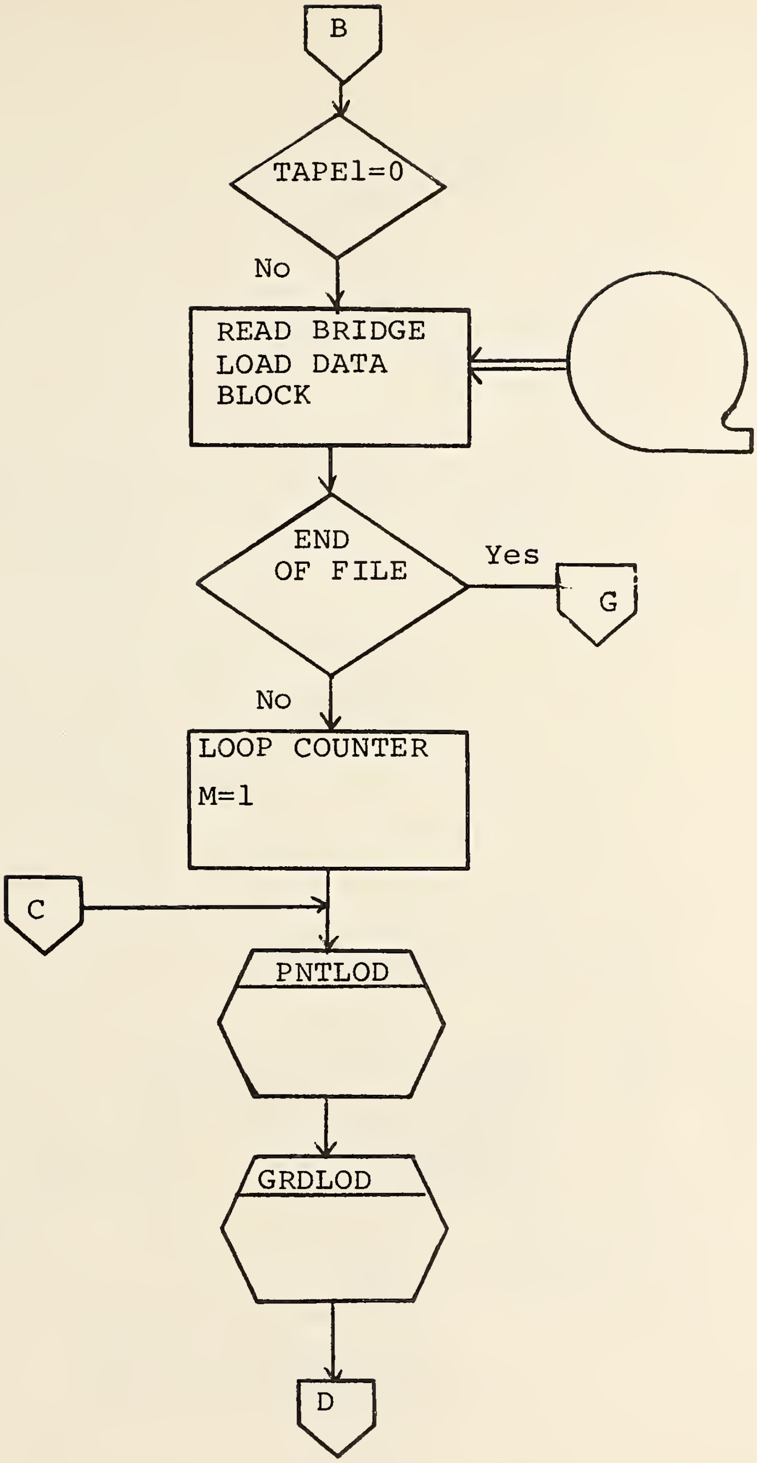


Figure 18. BREXEC Program Flow Chart (Continued)

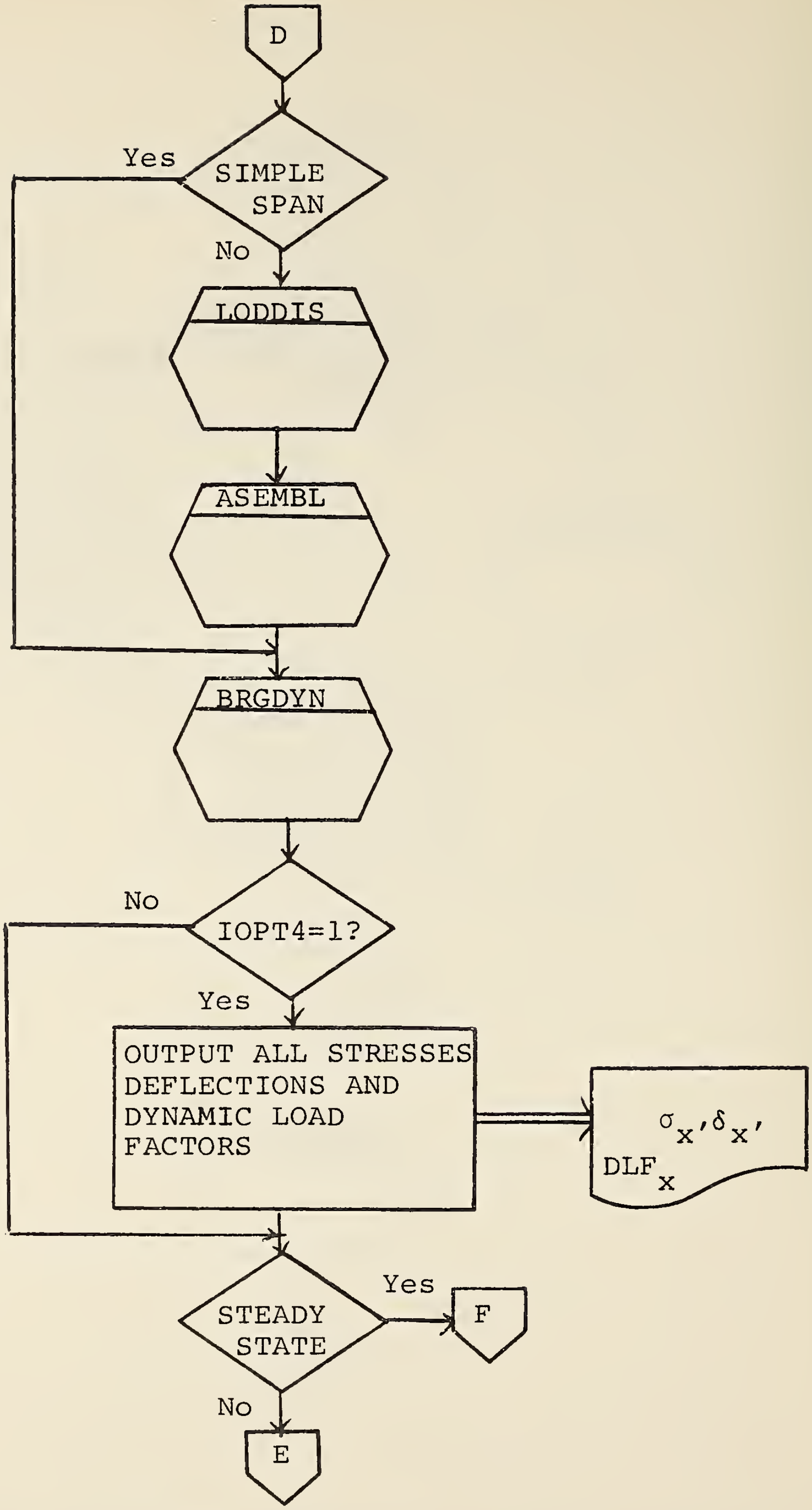


Figure 18. BREXEC Program Flow Chart (Continued)

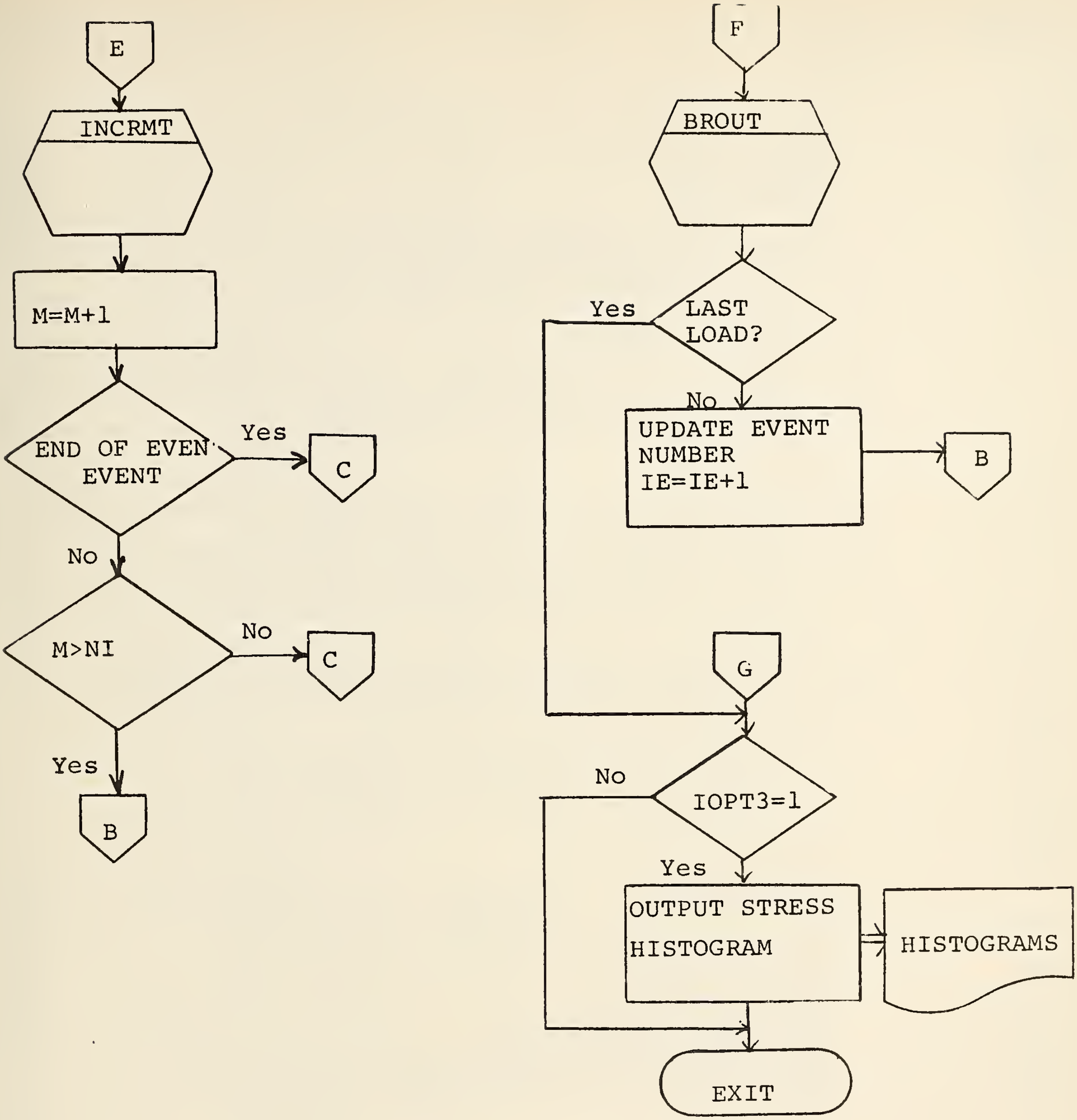


Figure 18. BREXEC Program Flow Chart (Continued)

```
               C
               C  BREXEC
               C
0001                 REAL NUC,NUS,IUR,ILR,IGO,IRGO,INRT,ILB,IT
0002                 REAL MD, MDT
0003                 LOGICAL LASTP
0004                 LOGICAL STEADY
0005                 INTEGER SIGC
               C
0006                 COMMON/BLK1/CLRL,CLRU, CLT, CRLD, CRUD, C2LRL, C2LRU, DR, DRT,
                    1   EC, ES, ILSC, NLRL, NLRU, T1, IL, NUC, NUS,
                    2   NRU(10), NRL(10), B(10), ISC(10), ALB(10), N2LRU(10),
                    3   N2LRL(10), T2(10), IT(10), IUR(10), ILR(10),ILB(10),
                    4   DELXP(10), ET(10),
                    5   AI(3,10), CI(3,10), CT(3,10), CCP(3,10), TCP(3,10), BCP(3,10),
                    6   DRG(3,10), C(3,10), IGO(3,10), INRT(3,10), EBAR(3,10),
                    7   BG(3,3,10), CG(3,3,10), TG(3,3,10), NGRU(3,3,10), NGRL(3,3,10),
                    8   CRU(3,3,10), CRL(3,3,10), IRGO(3,3,10)
                    9   ,DUM(1568)
               C
0007                 COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,
                    1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                    2   NXPTS, NDBUG(10),
                    3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                    4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                    6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                    7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
               C
0008                 COMMON /BLK3/ SUMHRX, DTL, IEVNTX, NOAXLX, NTRUKX, LAST,
                    1      LTYPE(20), WGTX(20),SPDX(20), LLANE(20), TIMEX(20),
                    2      XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
               C
0009                 COMMON /BLK5/ SIG(80),DFLN(80),DFL(80,10),ITIME,KTIME,
                    1      SIGP(120,3,10), SIGC(10,3,10), SIGTHS, DSIG
                    2      , IOPT1, IOPT2, IOPT3, IOPT4, NPTS(10), PTS(3,10)
                    3      , IPTS(3,10)
               C
0010                 COMMON /BLK6/ IBL(3,10)
               C
0011                 COMMON /BLK7/WGT,VEL,DELT,BLEN,NEV,NNL,NBEAM,NP(10),DMM(3),ITAPE2
                    1,SIGOUT(3,10)
               C
0012                 DIMENSION HISHED(20)
0013                 DIMENSION HEAD(17)
0014                 DIMENSION STRESI(10)
0015                 DIMENSION LOADBL(376)
0016                 EQUIVALENCE (HISHED(1),WGT)
0017                 EQUIVALENCE(LOADBL(1),SUMHRX)
0018                 NAMELIST/HISNAM/ WGT,VEL,DELT,BLEN,NEV,NNL,NBEAM,NP
0019                 DATA IFTST /0/
               C
0020                 CALL INIT
0021                 IO = 6
0022                 IN = 5
```

```
               C
               C   INITIALIZATION
               C
               C   INPUT PROGRAM OPTIONS
               C
0023                  READ (IN,1001) ITAPE1,ITAPE2,HEAD
0024             1001 FORMAT (    2I2, 17A4)
               C
0025                  WRITE (IO,1002) HEAD, ITAPE1, ITAPE2
0026             1002 FORMAT (1H1, 17A4,//,' INPUT TAPE NO.', I4,
                    1           '    OUTPUT TAPE NO.',I4)
               C
               C
0027                  KTIME = 1
0028                  ITIME = 1
0029                  NEV = 0
0030                  THRSH = 300.
0031                  SIGTHS = 300.0
0032                  DSIG = 3000.0
0033                  IEVENT = 0
               C
0034                  CALL INPTBD
0035                  CALL GRDNET
0036                  CALL MODULI
               C
0037                  IF(NSUP.EQ.2)  GO TO 50
0038                  CALL DMOMNT
0039                  GO TO 60
               C
0040               50 CALL STIFF(1)
               C
0041               60 IE = 1
               C
               C
0042                  LASTP = .FALSE.
               C
               C   PROGRAM MAIN LINE
               C
0043              100 CONTINUE
0044              130 CONTINUE
0045                  IF(ITAPE1.EQ.0)  GO TO 140
0046                  IF(ITAPE1.GT.5)  GO TO 135
0047                  READ (ITAPE1,1120,END=165) LOADBL
0048             1120 FORMAT(20A4)
0049                  GO TO 137
0050              135 READ(ITAPE1,END=165)  LOADBL
0051              137 CONTINUE
               C
0052                  IF (NDBUG(1).EQ.0)  GO TO 150
0053                  K = NOAXLX
0054                  WRITE(6,1145) SUMHRX
0055                  DO 146 I=1,K
0056              146 WRITE(6,1146) I,LNUMX(I), WEITX(I),XPOSX(I),DXPOSX(I),ACCLRX(I)
               C
```

```
 1145 FORMAT('OBRIDGE LOAD TIME =',F12.4,'SEC'//10X,'LANE',4X,'WEIGHT  '
     1,'POSITION',2X,'DISTANCE',2X,'ACCELERATION  ORDER'/)
 1146 FORMAT(7X,I2,3X,I2,F10.0,F10.2,F10.2,F14.2)
      WRITE (6,1050) IEVNTX, NTRUKX
 1050 FORMAT ('OEVENT NUMBER',I4,8X,'NO. OF TRUCKS',I4//,
     1  1X, 'TYPE    WEIGHT   SPEED    LANE    TIME ENTERING BRIDGE')
      DO 151 I=1,NTRUKX
  151 WRITE (6,1051) LTYPE(I),WGTX(I),SPDX(I),LLANE(I),TIMEX(I)
 1051 FORMAT(1X,I4,F10.0,F8.0,I8,F8.0)
  150 CONTINUE
      IF (IEVENT.NE.IEVNTX )  STEADY = .TRUE.
      IEVENT = IEVNTX
  140 CONTINUE
      IF (IFTST.GT.0) GO TO 200
      IFTST = IFTST + 1
      IF (DT.NE.0)    GO TO 90
C
      DIST = XSUP(2) - XSUP(1)
      IF(NSUP.EQ.2)  GO TO 80
      DO 75 I=3,NSUP
      DIS = XSUP(I) - XSUP(I-1)
      IF (DIS.LT.DIST)  DIST = DIS
   75 CONTINUE
   80 DT = DIST/1200.0
   90 CONTINUE
C
      XNI = DTL/DT + .5
      NI = XNI
      DT = DTL/NI
      IF(NDBUG(1).NE.0)  WRITE(IO,1200) DT,DTL,NI
 1200 FORMAT('0 DT  DTL  NI',2F10.3,I6)
C
      IF (ITAPE2.EQ.0)  GO TO 200
      DELT = DT
      BLEN = XLB
      NBEAM = NB
      DO 199 I=1,10
  199 NP(I) = NPTS(I)
C
  200 CONTINUE
C
      IF(ITAPE2.EQ.0)  GO TO 201
      IF (IEVNTX.EQ.NEV)  GO TO 202
      NEV = IEVNTX
      WGT = WEITX(1)
      VEL = SPDX(1)
      NNL = LNUMX(1)
C
C   CALCULATE NUMBER OF TIME INCREMENTS
C
      NJ = (XLB-1.0)/(DT*VEL)
C
      WRITE (ITAPE2) HISHED
      WRITE (IO,HISNAM)
```

```
0100                 GO TO 201
0101            202  NJ = NJ - NI
0102                 IF(NI.EQ.0)  GO TO 100
0103            201  CONTINUE
               C
0104                 M = 1
               C
0105            210  CONTINUE
0106                 CALL PNTLOD
0107                 IF (NAP.EQ.0.AND.STEADY)  GO TO 270
0108                 CALL GRDLOD
               C
0109                 IF(NSUP.GT.2)  CALL LODDIS
               C
0110                 IX4 = 0
0111                 DO 230 L=1,NSI
0112                 IX3 = IX4 + 1
0113                 IX4 = IX4 + IS(L)
0114                 IF (NSUP.EQ.2) GO TO 250
0115                 CALL ASEMBL(L)
0116            250  CONTINUE
0117                 CALL BRGDYN(L)
               C
               C
               C
0118                 IF (IOPT4.EQ.0)  GO TO 268
               C
               C   OUTPUT ALL STRESSES, DEFLECTIONS, AND DYNAMIC LOAD FACTORS
               C
               C   CONVERT DEFLECTIONS TO INCHES
               C
0119                 DO 261 I=1,NXPTS
0120            261  DFLN(I) = DFLN(I) * 12.0
               C
0121                 TIME2 = DT * (ITIME + KTIME - 1)
0122                 WRITE (IO,1261)  IE, IEVENT, TIME2
0123           1261  FORMAT (1H1,'STRESS EVENT',I4,'     LOAD EVENT',I4,'     TIME =',
                    1       F8.3)
               C
0124                 IX2 = 0
               C
0125                 DO 267 J=1,NB
0126                 IX1 = IX2 + 1
0127                 IX2 = IX2 + IS(L)
0128                 WRITE (IO,1262)  L,J
0129           1262  FORMAT (1H0,12X,'SPAN NO.',I3,'     BEAM NO.',I3/,1X,'POSITION')
               C
0130                 WRITE (IO,1263)  (X(I), I=IX3,IX4)
0131           1263  FORMAT (10F12.3)
0132                 WRITE (IO,1264)
0133           1264  FORMAT (1X,/,1X,'DYNAMIC BENDING STRESSES')
0134                 WRITE (IO,1263) (SIG(I),I=IX1,IX2)
               C
0135                 WRITE (IO,1266)
0136           1266  FORMAT ('0DYNAMIC DEFLECTIONS')
```

```
0137              WRITE(IO,1267) (DFL(I,L),I=IX1,IX2)
0138         1267 FORMAT (10F12.8)
            C
0139              WRITE (IO,1268)
0140         1268 FORMAT(1H0)
0141          267 CONTINUE
0142          230 CONTINUE
            C
0143          268 CONTINUE
0144              IF (ITAPE2.GT.0)  WRITE(ITAPE2) SIGOUT
            C
0145              IF (KTIME.LT.120)  GO TO 269
0146              CALL BROUT
0147              ITIME=ITIME+120
0148              KTIME = 0
0149          269 KTIME = KTIME + 1
0150              IF (ITAPE2.NE.0)  GO TO 270
            C
0151              IF (STEADY.AND.KTIME.GT.3)  GO TO 280
            C
0152          270 CONTINUE
0153              IF (NDBUG(1).EQ.0)  GO TO 271
0154              WRITE (IO,1271) M, NI, T, IE
0155         1271 FORMAT ('OINCREMENT M',I4,'  NI',I4,'   TIME',F12.3,
                 1      '      STRESS EVENT',I6)
0156          271 CONTINUE
0157              CALL INCRMT
0158              M = M + 1
0159              IF (ITAPE2.NE.0.AND.M.GT.NJ)  GO TO 280
0160              IF (M.GT.NI) GO TO 290
0161              GO TO 210
            C
0162          280 CONTINUE
0163              CALL BROUT
0164              IE = IE + 1
0165              KTIME = 1
0166              ITIME = 1
0167              IF (ITAPE2.NE.0.)  GO TO 100
0168              GO TO 270
0169          290 CONTINUE
0170              GO TO 100
            C
0171          165 CONTINUE
0172              IF (IOPT3.EQ.0)  GO TO 300
            C
            C   OUTPUT STRESS HISTOGRAMS
            C
0173              DO 164 I=1,10
0174          164 STRESI(I) = DSIG * I
0175              WRITE (IO,1165) (STRESI(I),I=1,10)
0176         1165 FORMAT ('1   STRESS HISTOGRAMS',//,' 10% INTERVALS',10(' TO',F8.1)
                 1 ,//,' BEAM POINT')
            C
0177              DO 367 J=1,NB
```

```
0178                NPT = NPTS(J)
0179                IF(NPT.EQ.0)  GO TO 367
           C
0180                DO 167 K=1,NPT
0181            167 WRITE (IO,1167) J,PTS(K,J), (SIGC(I,K,J),I=1,10)
0182           1167 FORMAT (1X,I4, F10.3, 10(5X,I6))
0183            367 CONTINUE
           C
0184            168 CONTINUE
0185            300 STOP 07
           C
0186                END
```

INIT

The subroutine INIT initializes common blocks BLK1 and BLK2 to zero.

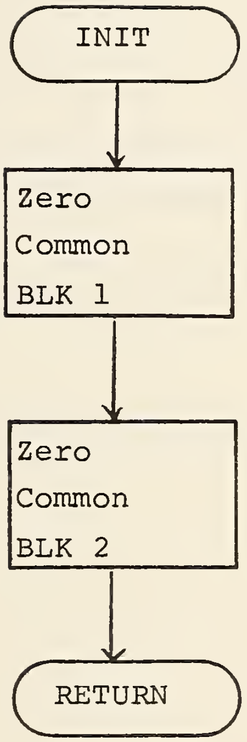


Figure 19. INIT Program Flow Chart

```
      SUBROUTINE INIT
C
C   SET DATA BLOCKS TO ZERO
C
      COMMON /BLK1/ IDM1(6400)
      COMMON /BLK2/ IDM2(4005)
      COMMON /BLK4/ IDM4(19520)
      COMMON /BLK5/ IDM5(4938)
      COMMON /BLK7/ IDM7(51)
      DO 10 I=1,1208
   10 IDM1(I) = 0
C
      DO 20 I=1,4005
   20 IDM2(I) = 0
C
      DO 40 I=1,19520
   40 IDM4(I) = 0
C
      DO 30 I=1,4938
   30 IDM5(I) = 0
C
      DO 70 I=1,51
   70 IDM7(I) = 0
C
      RETURN
      END
```

INPTBD

The INPTBD reads all input bridge data as:

Single element data
Longitudinal beam data
Lateral beam data
Output specifications

All elements are printed out after input. All data input in inches is converted to feet for computation.

A detailed list of input data is found in the section Data Input above.

No subroutines are called by this routine.

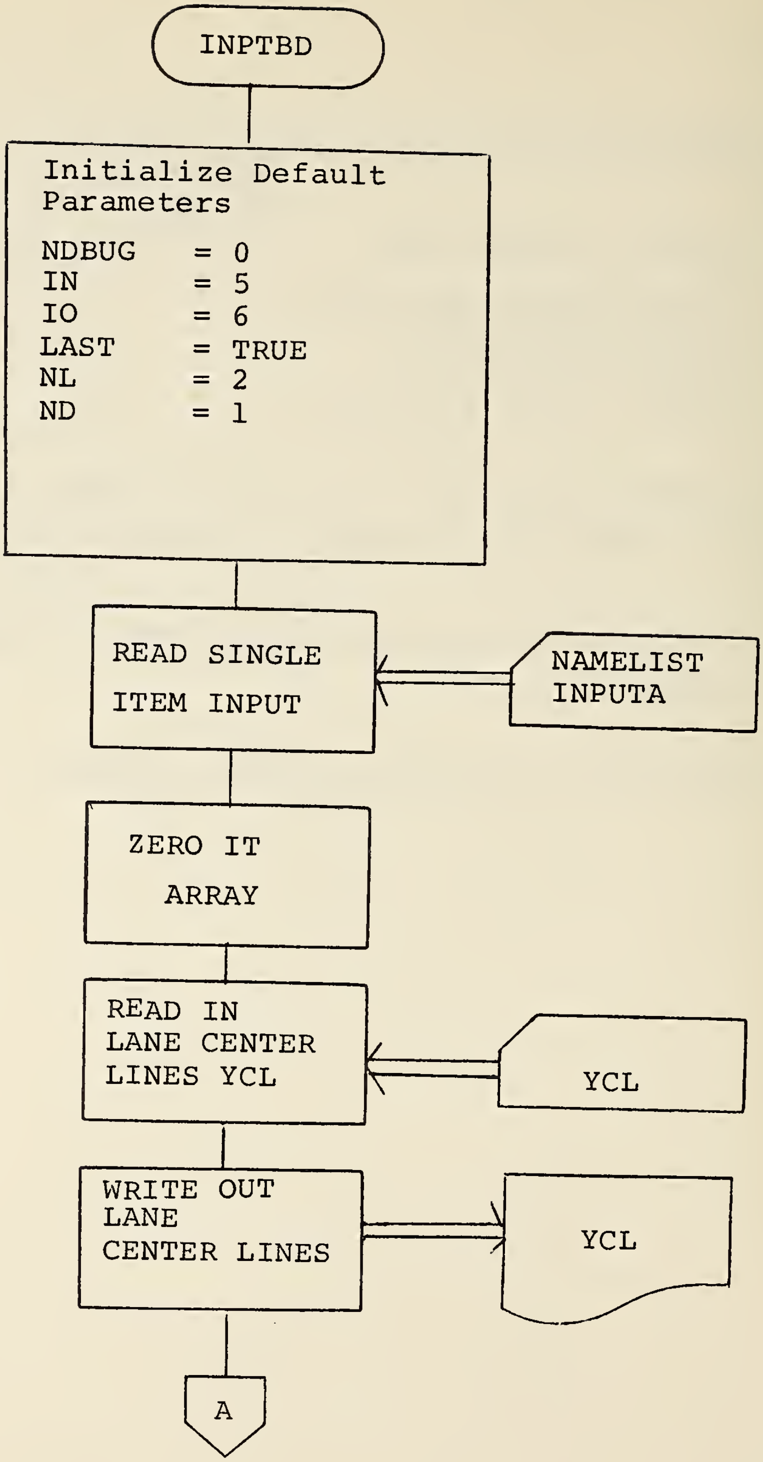


Figure 20. INPTBD Program Flow Chart

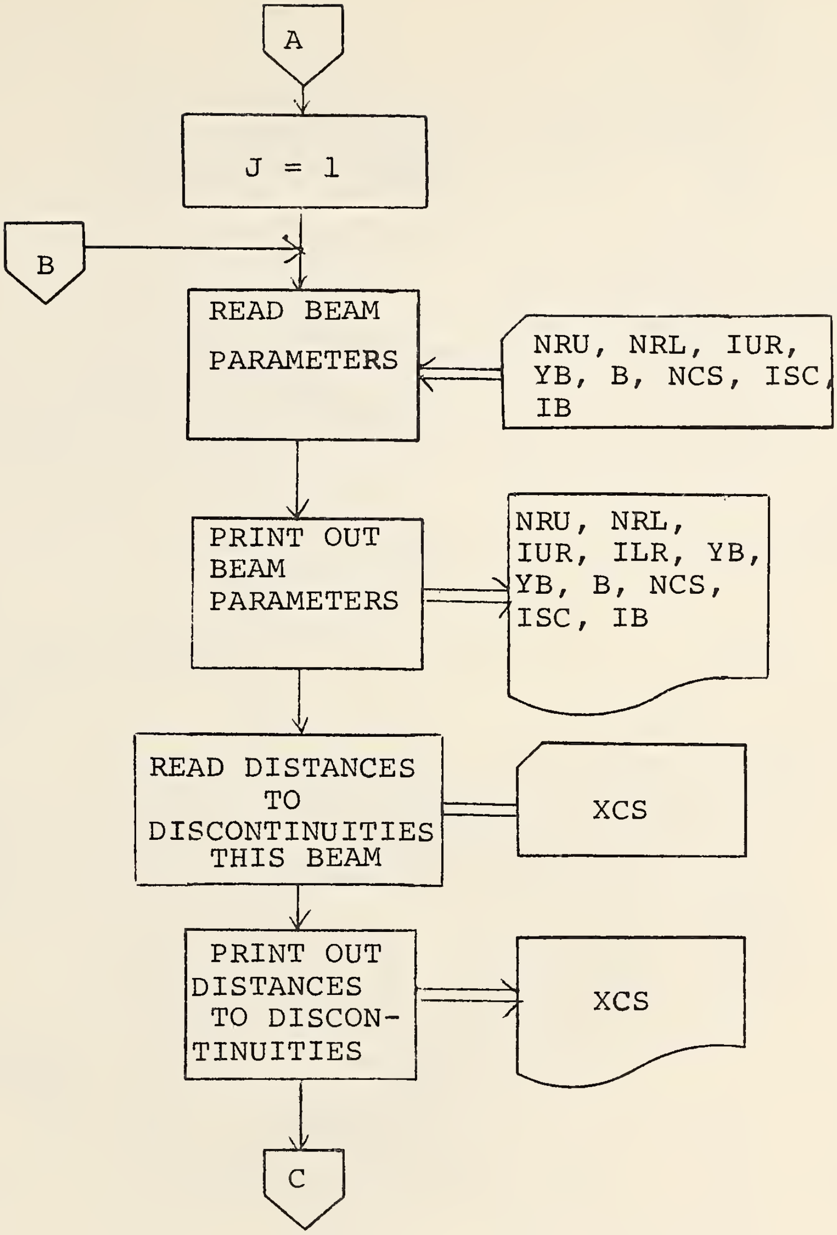

A
J = 1
B
READ BEAM PARAMETERS
NRU, NRL, IUR, YB, B, NCS, ISC, IB
PRINT OUT BEAM PARAMETERS
NRU, NRL, IUR, ILR, YB, YB, B, NCS, ISC, IB
READ DISTANCES TO DISCONTINUITIES THIS BEAM
XCS
PRINT OUT DISTANCES TO DISCONTINUITIES
XCS
C

Figure 20. INPTBD Program Flow Chart (Continued)

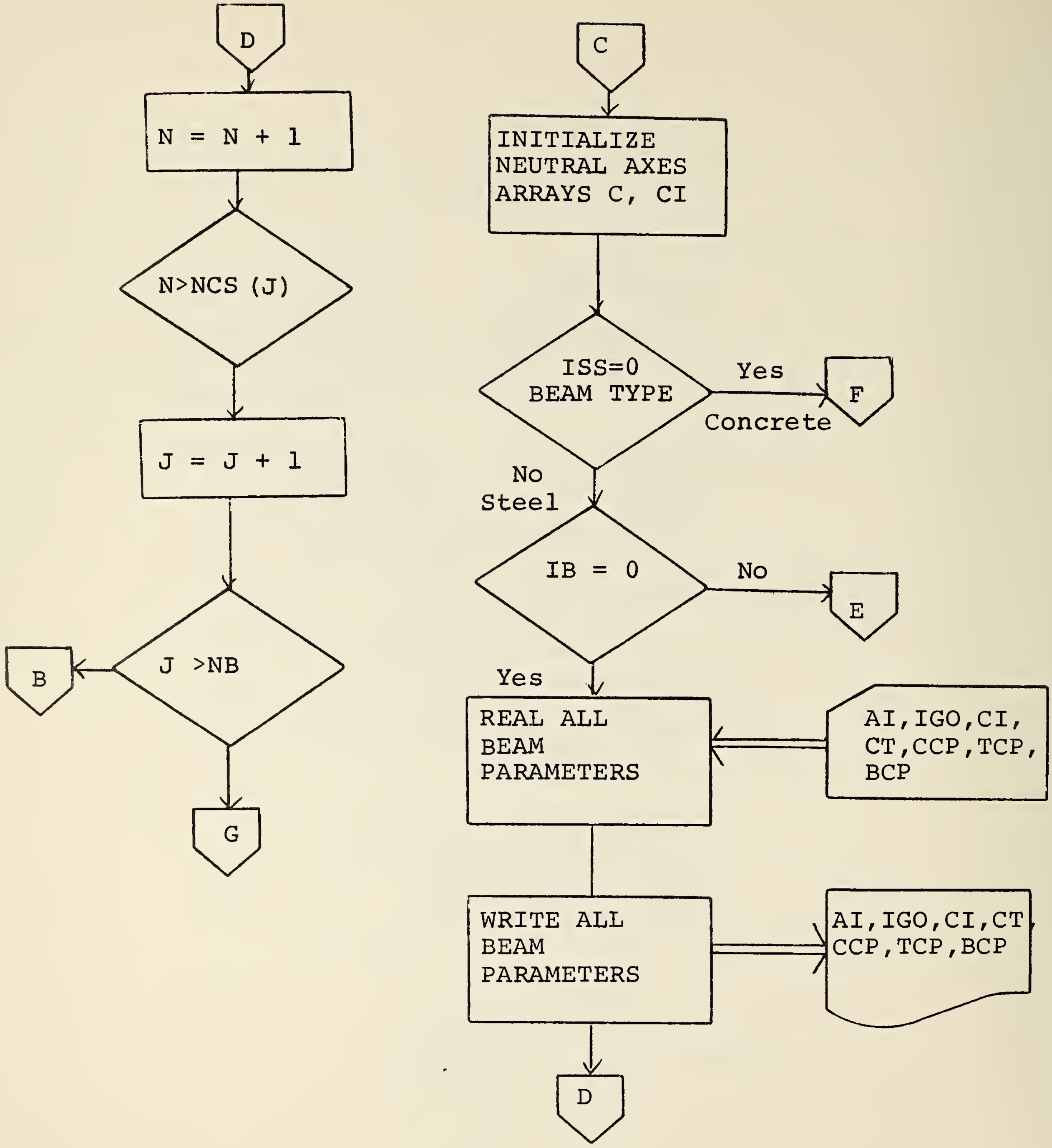

D
N = N + 1
N>NCS (J)
J = J + 1
J >NB
B
G
C
INITIALIZE NEUTRAL AXES ARRAYS C, CI
ISS=0 BEAM TYPE
Yes
Concrete
F
No
Steel
IB = 0
No
E
Yes
REAL ALL BEAM PARAMETERS
AI,IGO,CI, CT,CCP,TCP, BCP
WRITE ALL BEAM PARAMETERS
AI,IGO,CI,CT, CCP,TCP,BCP
D

Figure 20. INPTBD Program Flow Chart (Continued)

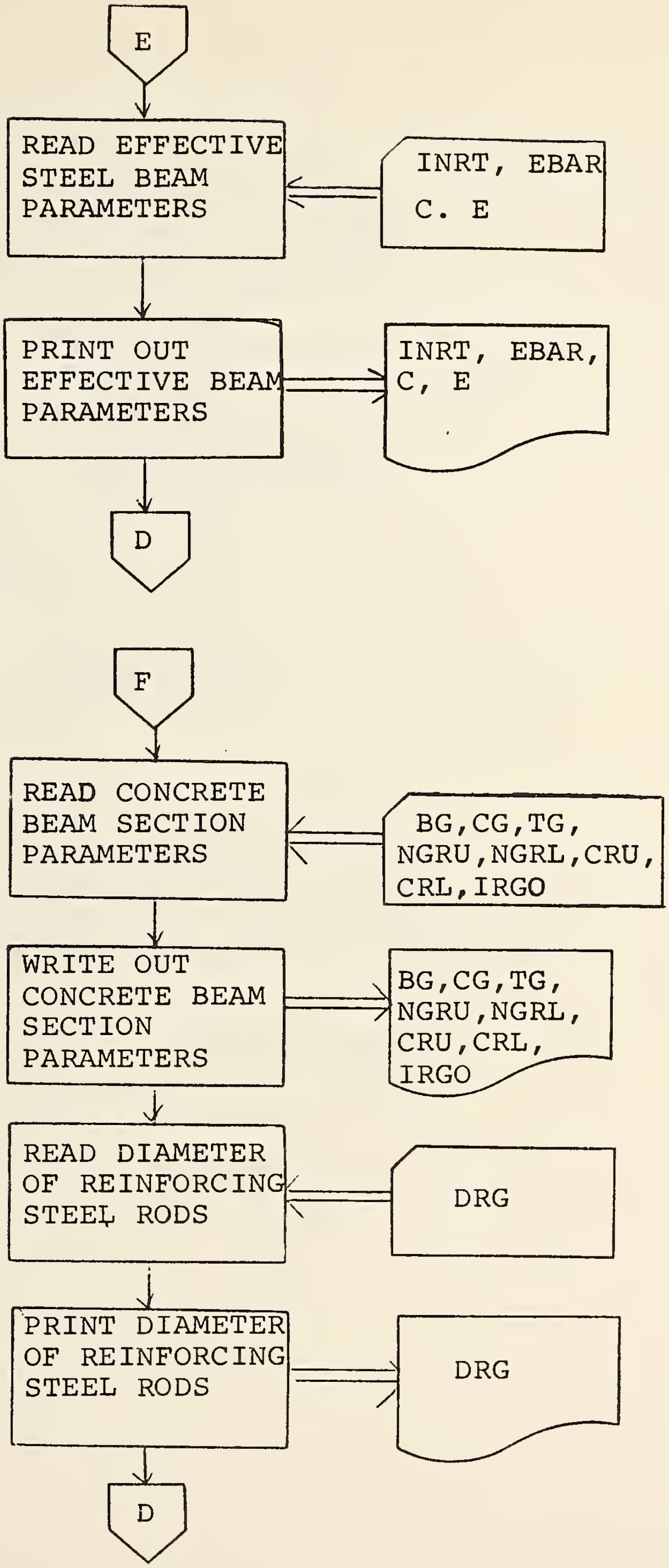

E
READ EFFECTIVE STEEL BEAM PARAMETERS
INRT, EBAR C. E
PRINT OUT EFFECTIVE BEAM PARAMETERS
INRT, EBAR, C, E
D
F
READ CONCRETE BEAM SECTION PARAMETERS
BG,CG,TG, NGRU,NGRL,CRU, CRL,IRGO
WRITE OUT CONCRETE BEAM SECTION PARAMETERS
BG,CG,TG, NGRU,NGRL, CRU,CRL, IRGO
READ DIAMETER OF REINFORCING STEEL RODS
DRG
PRINT DIAMETER OF REINFORCING STEEL RODS
DRG
D

Figure 20. INPTBD Program Flow Chart (Continued)

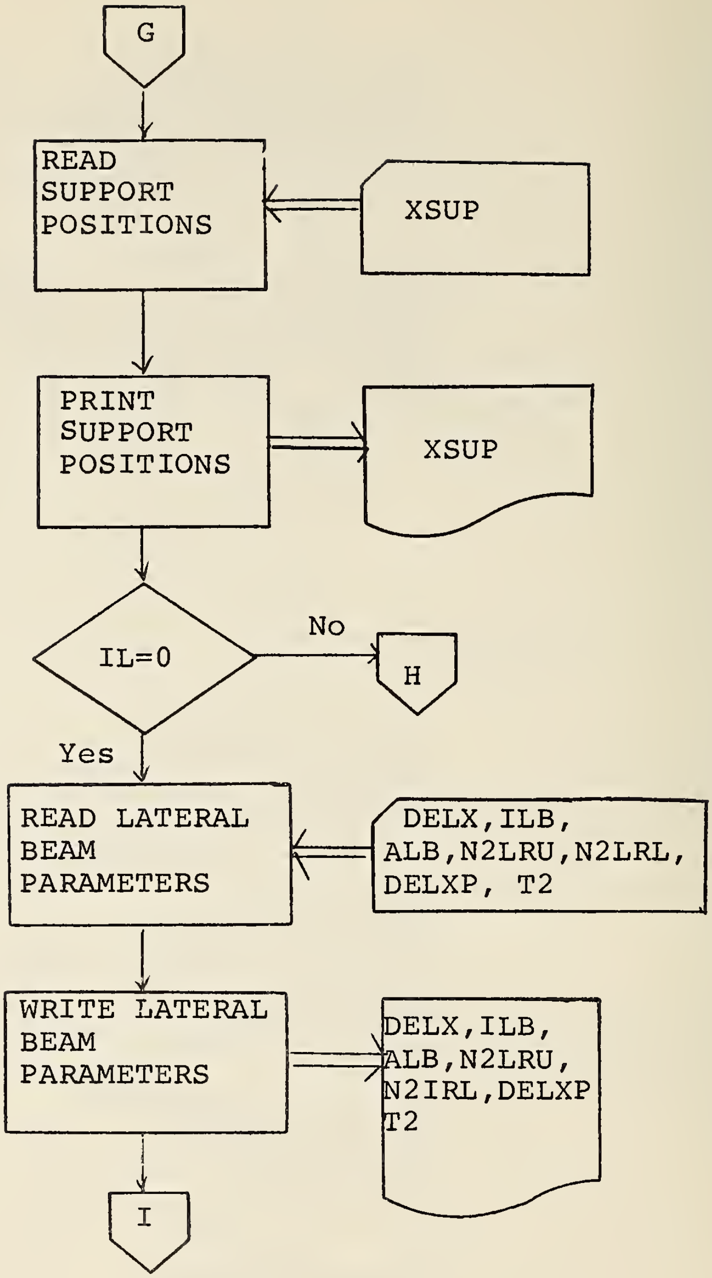

G
READ SUPPORT POSITIONS
XSUP
PRINT SUPPORT POSITIONS
XSUP
IL=0
No
H
Yes
READ LATERAL BEAM PARAMETERS
DELX,ILB, ALB,N2LRU,N2LRL, DELXP, T2
WRITE LATERAL BEAM PARAMETERS
DELX,ILB, ALB,N2LRU, N2IRL,DELXP T2
I

Figure 20. INPTBD Program Flow Chart (Continued)

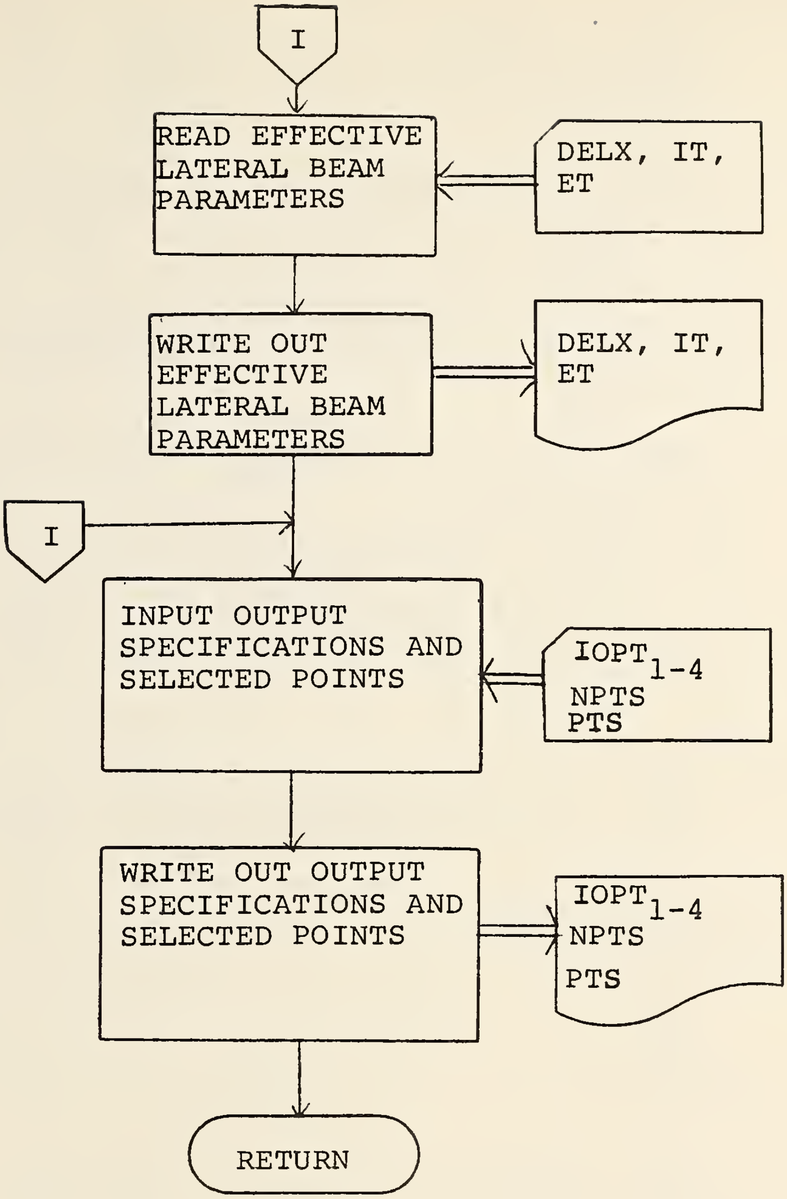

I
READ EFFECTIVE LATERAL BEAM PARAMETERS
DELX, IT, ET
WRITE OUT EFFECTIVE LATERAL BEAM PARAMETERS
DELX, IT, ET
I
INPUT OUTPUT SPECIFICATIONS AND SELECTED POINTS
IOPT1-4
NPTS
PTS
WRITE OUT OUTPUT SPECIFICATIONS AND SELECTED POINTS
IOPT1-4
NPTS
PTS
RETURN

Figure 20. INPTBD Program Flow Chart (Continued)

```
0001                 SUBROUTINE INPTBD
0002                 REAL NUC,NUS,IUR,ILR,IGO,IRGO,INRT,ILB,IT
0003                 REAL MD, MDT
0004                 LOGICAL LAST
           C
0005                 COMMON/BLK1/CLRL,CLRU, CLT, CRLD, CRUD, C2LRL, C2LRU, DR, DRT,
                    1   EC, ES, ILSC, NLRL, NLRU, T1, IL, NUC, NUS,
                    2   NRU(10), NRL(10), B(10), ISC(10), ALB(10), N2LRU(10),
                    3   N2LRL(10), T2(10), IT(10), IUR(10), ILR(10),ILB(10),
                    4   DELXP(10), ET(10),
                    5   AI(3,10), CI(3,10), CT(3,10), CCP(3,10), TCP(3,10), BCP(3,10),
                    6   DRG(3,10), C(3,10), IGO(3,10), INRT(3,10), EBAR(3,10),
                    7   BG(3,3,10), CG(3,3,10), TG(3,3,10), NGRU(3,3,10), NGRL(3,3,10),
                    8   CRU(3,3,10), CRL(3,3,10), IRGO(3,3,10)
                    9   ,DUM(1568)
           C
0006                 COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     (
                    1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                    2   NXPTS, NDBUG(10),
                    3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                    4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                    6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                    7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
           C
0007                 COMMON /BLK5/ SIG(80),DFLN(80),DFL(80,10),ITIME,KTIME,
                    1     SIGP(120,3,10), SIGC(10,3,10), SIGTHS, DSIG
                    2     , IOPT1, IOPT2, IOPT3, IOPT4, NPTS(10), PTS(3,10)
                    3     , IPTS(3,10)
           C
0008                 NAMELIST/INPUTA/XLB,NL,YBCL,   EC,ES,NUC,NUS,DT,THRSH, WEIGHT,
                    1  DR,    CRUD,CRLD,NB,NSUP,C2LRU,C2LRL,DRT,NLRU,NLRL,CLRU,CLRL,
                    2  ILSC, T1, IL, SIGTHS, DSIG, NDBUG
           C
0009                 DIMENSION IB(10)
           C
0010                 DO 20 I=1,10
0011              20 NDBUG(I) = 0
           C
0012                 IN=5
0013                 IO=6
0014                 LAST=.TRUE.
0015                 NL = 2
0016                 ND = 1
           C
0017                 READ (IN,INPUTA)
0018                 WRITE (IO,INPUTA)
           C
           C  CONVERT UNITS TO FEET
           C
0019                 EC = EC*144.
0020                 ES = ES*144.
0021                 DR = DR/12.
0022                 CRUD = CRUD/12.
0023                 CRLD = CRLD/12.
```

```
0024                   C2LRU = C2LRU/12.
0025                   C2LRL = C2LRL/12.
0026                   DRT = DRT/12.
0027                   CLRU = CLRU/12.
0028                   CLRL = CLRL/12.
0029                   T1 = T1/12.
                 C
0030                   DO 100 L=1, 10
0031               100 IT(L)=0.
                 C
                 C   READ CENTER LINE POSITIONS
                 C
0032                   READ (IN,1000) (YCL(J),J=1,NL)
0033              1000 FORMAT (2F10.4)
0034                   WRITE (IO,2000)
0035              2000 FORMAT (1H1,'YCL(J)')
0036                   WRITE (IO,2001) (YCL(J),J=1,NL)
0037              2001 FORMAT (1X,2F16.4)
                 C
                 C   READ SUPPORT POSITIONS
                 C
0038                   READ(IN,3015) (XSUP(L),L=1,NSUP)
                 C
0039                   WRITE (IO,2016) (XSUP(L),L=1,NSUP)
0040              2016 FORMAT ('0SUPPORT POSITIONS',11F10.2)
0041              3015 FORMAT (11F7.2)
                 C
                 C   READ BEAM PARAMETERS
                 C
0042                   DO  1 J=1,NB
0043                   READ (IN,1001) NRU(J),NRL(J),IUR(J),ILR(J),YB(J),B(J),NCS(J),ISC(J
                      1),IB(J)
0044              1001 FORMAT (2I3,2E10.3,2F8.3,3I1)
0045                   WRITE (IO,2002)
0046              2002 FORMAT ('0  BEAM    NRU    NRL                IUR                ILR',
                      1    '          YB             B    NCS    ISC     IB')
0047                   WRITE (IO,2003) J,NRU(J),NRL(J),IUR(J),ILR(J),YB(J),B(J),NCS(J),
                      1   ISC(J), IB(J)
0048              2003 FORMAT (3I6,2E16.3,2F12.3,3I6)
                 C
                 C   CONVERT UNITS TO FEET
                 C
0049                   IUR(J) = IUR(J)/20736.0
0050                   ILR(J) = ILR(J)/20736.0
                 C
0051                   NCSJ=NCS(J)
0052                   READ (IN,1002) (XCS(N,J),N=1,NCSJ)
0053              1002 FORMAT (3F8.3)
0054                   WRITE (IO,2004)
0055              2004 FORMAT (1X,'XCS(N,J)')
0056                   WRITE (IO,2005) (XCS(N,J),N=1,NCSJ)
0057              2005 FORMAT (3F12.3)
                 C
0058                   DO 2 N=1,NCSJ
```

```
0059                CI(N,J) = 0.0
0060                C(N,J) = 0.0
0061                IF (IB(J).NE.0) GO TO 5
              C
              C   READ ALL BEAM PARAMETERS
              C
0062                IF (ISC(J).EQ.0) GO TO 3
              C
              C   STEEL BEAM
              C
0063                READ (IN,1003) AI(N,J),IGO(N,J),CI(N,J),CT(N,J),CCP(N,J)
                   1   , TCP(N,J), BCP(N,J)
0064           1003 FORMAT (7F8.3)
0065                WRITE (IO,2006)
0066           2006 FORMAT ('0          AI          IGO          CI          CT'
                   1 ,'          CCP          TCP          BCP')
0067                WRITE (IO,2007) AI(N,J),IGO(N,J),CI(N,J),CT(N,J),CCP(N,J)
                   1 ,TCP(N,J), BCP(N,J)
0068           2007 FORMAT (1X,7F12.3)
              C
              C   CONVERT UNITS TO FEET
              C
0069                AI(N,J) = AI(N,J)/144.0
0070                IGO(N,J) = IGO(N,J)/20736.0
0071                CI(N,J) = CI(N,J)/12.0
0072                CT(N,J) = CT(N,J)/12.0
0073                CCP(N,J) = CCP(N,J)/12.0
0074                TCP(N,J) = TCP(N,J)/12.0
              C
0075                GO TO 2
              C
              C   READ EFFECTIVE BEAM PARAMETERS
              C
0076              5 READ (IN,1007) INRT(N,J),EBAR(N,J),C(N,J),E(N,J)
0077           1007 FORMAT (2E10.3,F8.3,E10.3)
0078                WRITE (IO,2014)
0079           2014 FORMAT (1X,'INRT,EBAR,C,E')
0080                WRITE (IO,2015) INRT(N,J),EBAR(N,J),C(N,J),E(N,J)
0081           2015 FORMAT (1X,2E16.3,F12.3,E16.3)
0082                INRT(N,J) = INRT(N,J)/20736.0
0083                EBAR(N,J) = EBAR(N,J)*144.0
0084                C(N,J) = C(N,J)/12.0
0085                E(N,J) = E(N,J)*144.0
              C
0086                GO TO 2
              C
              C   CONCRETE
              C
0087              3 CONTINUE
0088                WRITE (IO,2010)
0089           2010 FORMAT ('0          BG          CG          TG   NGRU   NGRL',
                   1   '          CRU          CRL          IRGO')
              C
0090                DO 4 M=1,3
```

```
0091              READ (IN,1005) BG(M,N,J),CG(M,N,J),TG(M,N,J),NGRU(M,N,J),NGRL(M,N,
                1J),CRU(M,N,J),CRL(M,N,J),IRGO(M,N,J)
0092         1005 FORMAT (3F8.3,2I3,3F8.3)
0093              WRITE (IO,2011) BG(M,N,J),CG(M,N,J),TG(M,N,J),NGRU(M,N,J),NGRL(M,N
                1,J),CRU(M,N,J),CRL(M,N,J),IRGO(M,N,J)
0094         2011 FORMAT (1X,3F12.3,2I6,3F12.3)
            C
            C   CONVERT UNITS TO FEET
            C
0095              BG(M,N,J) = BG(M,N,J)/12.0
0096              CG(M,N,J) = CG(M,N,J)/12.0
0097              TG(M,N,J) = TG(M,N,J)/12.0
0098              CRU(M,N,J) = CRU(M,N,J)/12.0
0099              CRL(M,N,J) = CRL(M,N,J)/12.0
0100              IRGO(M,N,J) = IRGO(M,N,J)/20736.0
            C
0101            4 CONTINUE
            C
0102              READ (IN,1006) DRG (N,J)
0103         1006 FORMAT (F8.3)
0104              WRITE (IO,2012)
0105         2012 FORMAT (1X,'DRG')
0106              WRITE (IO,2013) DRG(N,J)
0107         2013 FORMAT (1X,F12.3)
            C
            C   CONVERT UNITS TO FEET
            C
0108              DRG(N,J) = DRG(N,J)/12.0
            C
0109            2 CONTINUE
0110            1 CONTINUE
            C
            C
0111              IF(IL.EQ.0)  WRITE (IO,2018)
0112              IF(IL.NE.0)  WRITE (IO,2020)
0113         2018 FORMAT ('0',          '          DELX           ILB                 ALB',
                1     '  N2LRU   N2LRL      DELXP          T2')
0114         2020 FORMAT ('0',          '          DELX              IT                ET')
0115              NSI=NSUP-1
0116              DO 6 L=1,NSI
0117              IF (IL   .NE.0) GO TO 7
0118              READ (IN,1009)             DELX(L),
                1                  ILB(L),ALB(L),N2LRU(L),N2LRL(L),DELXP(L),T2(L)
0119         1009 FORMAT(2F8.3,E10.3,      2I3,2F8.3)
0120              WRITE (IO,2019)             DELX(L),
                1                   ILB(L),ALB(L),N2LRU(L),N2LRL(L),DELXP(L),T2(L)
0121         2019 FORMAT (1X,1F12.3,E16.3,F12.3,2I6,2F12.3)
0122              IT(L) = 0.
            C
            C   CONVERT UNITS TO FEET
            C
0123              ALB(L) = ALB(L)/144.0
0124              ILB(L) = ILB(L)/20736.0
0125              DELXP(L) = DELXP(L)/12.0
```

```
           T2(L) = T2(L)/12.0
C
           GO TO 6
C
         7 READ (IN,1010)         DELX(L), IT(L), ET(L)
      1010 FORMAT ( F8.3,2E10.3)
           WRITE (IO,2021)         DELX(L),IT(L),ET(L)
      2021 FORMAT (1X, F12.3,2E16.3)
C
C   CONVERT UNITS TO FEET
C
           IT(L) = IT(L)/20736.0
           ET(L) = ET(L)*144.0
C
         6 CONTINUE
C
C   READ OUTPUT SPECIFICATIONS
C
           READ (IN,3001) IOPT1, IOPT2, IOPT3, IOPT4
      3001 FORMAT (4I1)
           WRITE (IO,3002) IOPT1, IOPT2, IOPT3, IOPT4
      3002 FORMAT (1H0, 'IOPT1   IOPT2   IOPT3   IOPT4',/,1X,4I6)
C
           IF (IOPT1.EQ.0.AND.IOPT2.EQ.0.AND.IOPT3.EQ.0)  GO TO 60
C
           DO 50 I=1,NB
        50 READ (IN,3004) NPTS(I), (PTS(J,I),J=1,3)
      3004 FORMAT (I1,3F8.3)
C
           WRITE(IO,3005)
      3005 FORMAT ('ONPTS      POINT 1      POINT 2      POINT 3   ',
     1     '   INPUT  POINTS'//)
C
           DO 55 I=1,NB
           NJ = NPTS(I)
        55 WRITE (IO,3006)  NPTS(I), (PTS(J,I),J=1,NJ)
      3006 FORMAT (1X, I4, 3F12.3)
        60 CONTINUE
C
           RETURN
           END
```

GRDNET

The GRDNET routine establishes the grid point values X and the number of longitudinal grid points per span IS(L).

The number of points is first established

$$IS_L = \frac{XSUP_{L+1} - XSUP_2}{\Delta X_L}$$

If $IS_L NB$ is greater than 80, IS_L is recalculated to

$$IS_L = \frac{80}{NB}$$

and a new grid incremental distance is calculated

$$\Delta X_L = \frac{XSUP_{L+1} - XSUP_L}{IS_L}$$

A grid weight is calculated as

$$\Delta W_L = \frac{\left(\dfrac{XSUP_{L+1} - XSUP_2}{XLB - Weight}\right)}{\left(\dfrac{IS_L NB}{G}\right)}$$

Grid positions are calculated as

$$X_i = XSUP_L + \frac{\Delta X}{2} + (i-1)\ \Delta X_L$$

Output points are converted to the nearest grid point.

No subroutines are called by this routine.

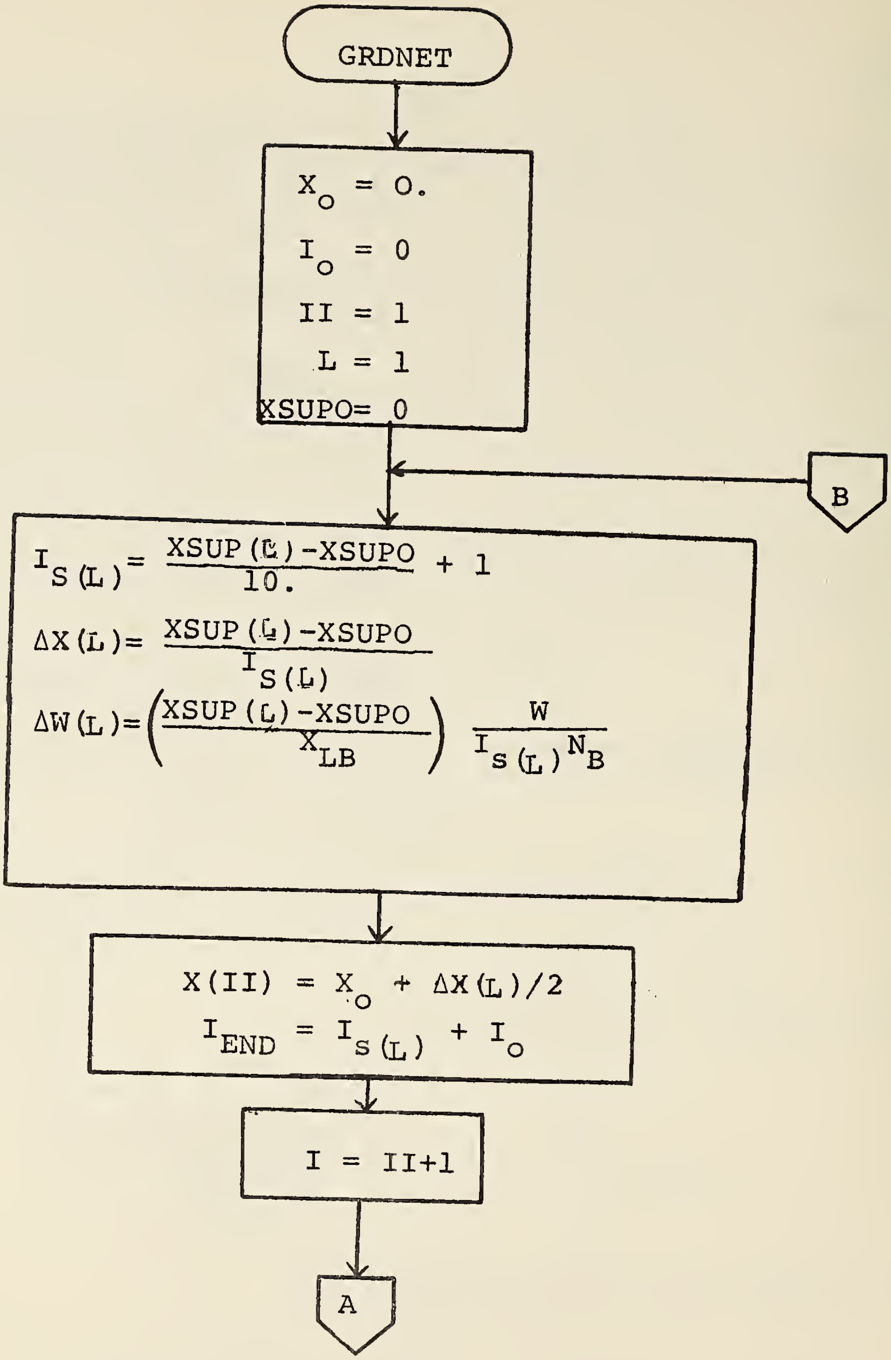

GRDNET
Xo = 0.
Io = 0
II = 1
L = 1
XSUPO= 0
B
IS(L) = (XSUP(L)-XSUPO)/10. + 1
ΔX(L) = (XSUP(L)-XSUPO)/IS(L)
ΔW(L) = ((XSUP(L)-XSUPO)/XLB) W/(IS(L) NB)
X(II) = Xo + ΔX(L)/2
IEND = IS(L) + Io
I = II+1
A

Figure 21. GRDNET Program Flow Chart

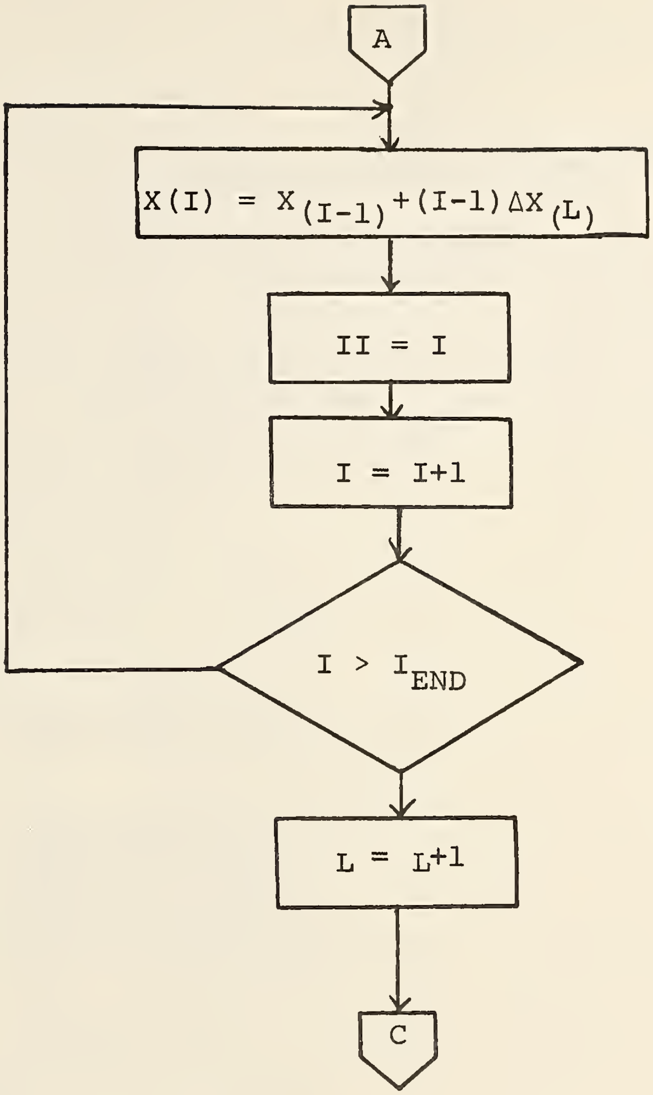


Figure 21. GRDNET Program Flow Chart (Continued)

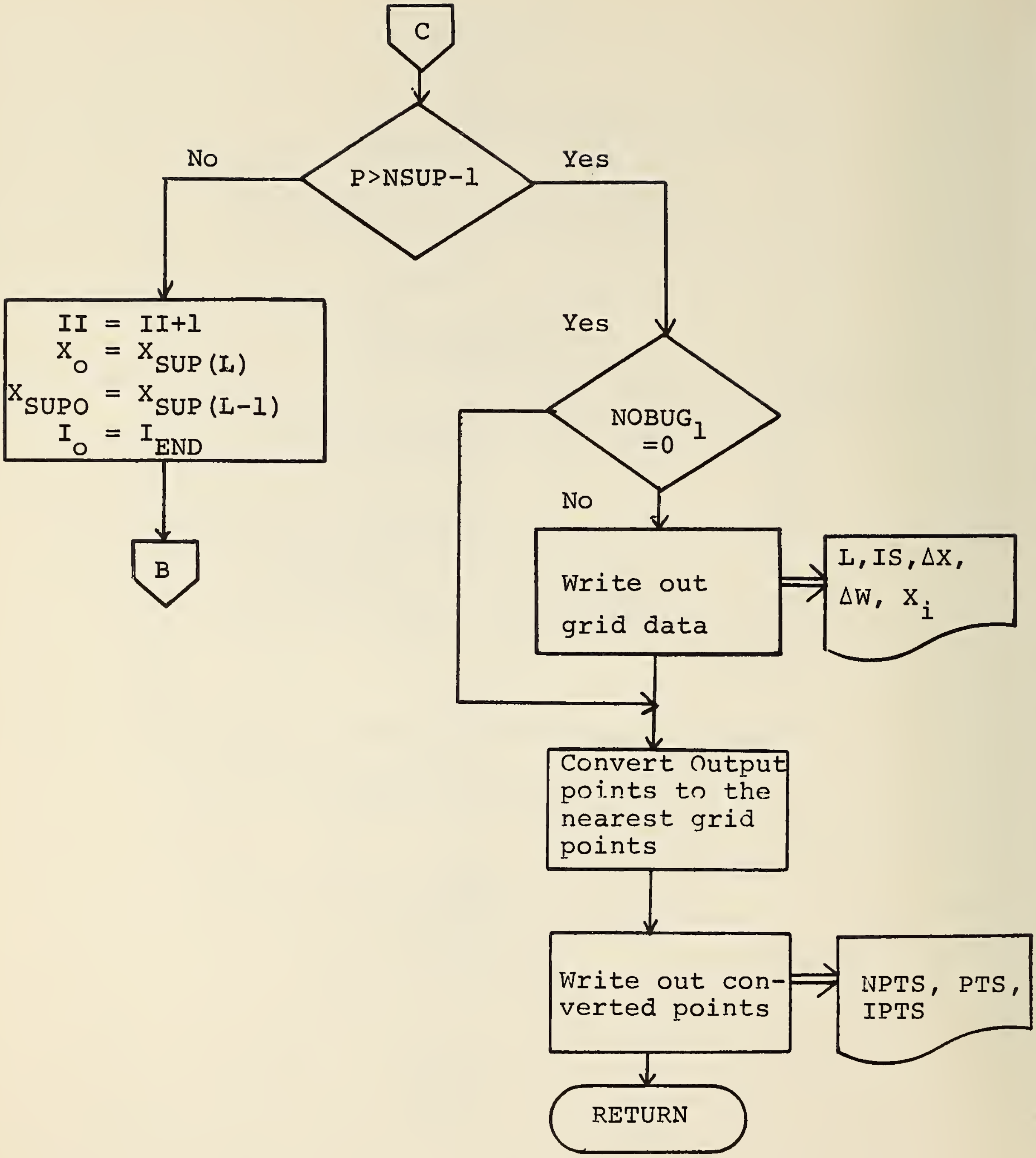


Figure 21. GRDNET Program Flow Chart (Continued)

```
0001                SUBROUTINE GRDNET
              C
              C   THIS ROUTINE ESTABLISHES X GRID POINTS AND ISL
              C
0002                REAL MD, MDT
              C
0003                COMMON /BLK2/ DT,       THRSH,  IE, IEVENT, IN, IO, ISS, ITER,
                  1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                  2   NXPTS, NDBUG(10),
                  3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                  4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                  6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                  7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
              C
0004                COMMON /BLK5/ SIG(80),DFLN(80),DFL(80,10),ITIME,KTIME,
                  1      SIGP(120,3,10), SIGC(10,3,10), SIGTHS, DSIG
                  2      , IOPT1, IOPT2, IOPT3, IOPT4, NPTS(10), PTS(3,10)
                  3      , IPTS(3,10)
              C
              C
0005                DO 20 I=1,100
0006             20 X(I) = 0.0
              C
0007                G = 32.2
0008                XO = 0.
0009                IO = 0
0010                II = 1
              C
0011                DO 1 L=1,NSI
0012                DXSUP =  XSUP(L+1) - XSUP(L)
0013                IS(L) = DXSUP/DELX(L)
              C
0014                ITST =  IS(L)*NB
0015                IF(ITST.LE.80)  GO TO 14
0016                ITST = 80/NB
0017                DELX(L) = DXSUP/ITST
0018                IS(L) = ITST
0019                WRITE (IO,100)  DELX(L),L
0020            100 FORMAT ('OINPUT DELX TOO SMALL - DELX CHANGED TO',F9.3,
                  1  '     FOR L = ',I2)
              C
0021             14 CONTINUE
0022                DELW(L) = DXSUP/XLB*WEIGHT/(IS(L)*NB)/G
              C
0023                X(II) = XO + DELX(L)/2.
0024                IEND = IS(L) + IO
              C
0025                IL = II + 1
0026                DO 2 I=IL,IEND
0027                X(I) = X(I-1) +DELX(L)
0028              2 II = I
0029                II = II + 1
0030                XO = XSUP(L+1)
0031                IO = IEND
```

```
0032              1 CONTINUE
            C
0033                IF (NDBUG(1).EQ.0)  GO TO 15
0034                WRITE (IO,10)
0035             10 FORMAT (   'OGRDNET')
0036                DO 3 L=1,NSI
0037              3 WRITE (IO,11) L,IS(L), DELX(L), DELW(L), XSUP(L), XSUP(L+1)
0038             11 FORMAT (1X,'NO. SPANS=',I3,2X,'NO. GRIDS=',I3,'  DELX=',E12.4,
                  1   '  DELW=',E12.4,'  SUPPORTS  ',2E10.3)
0039             15 CONTINUE
0040                WRITE (IO,12)
0041             12 FORMAT (   '0 GRID CENTERS LONGITUDINALLY = X')
0042                WRITE (IO,13) X
0043             13 FORMAT (1X,10E12.4)
            C
            C
            C    CALCULATE TOTAL X POINTS
            C
0044                NXPTS = 0
0045                DO 18 I=1,NSI
0046                NXPTS = NXPTS + IS(L)
0047             18 CONTINUE
            C
            C
            C   CONVERT SAMPLE POINTS TO NEAREST GRID POINT
            C
0048                IF (IOPT1.EQ.0.AND.IOPT2.EQ.0.AND.IOPT3.EQ.0)  GO TO 60
            C
0049                DO 30 J=1,NB
0050                NPT = NPTS(J)
0051                IF (NPT.EQ.0)  GO TO 30
0052                DO 25 I=1,NPT
0053                DO 21 K=1,NXPTS
0054                IF (PTS(I,J).EQ.X(K))  GO TO 24
0055                IF (PTS(I,J).EQ.X(K).OR.PTS(I,J).GT.X(K+1))  GO TO 21
0056                GO TO 22
0057             21 CONTINUE
0058             22 XT1 = PTS(I,J) - X(K)
0059                XT2 = X(K+1) - PTS(I,J)
0060                IF (XT2.GT.XT1)  GO TO 23
0061                PTS(I,J) = X(K+1)
0062                IPTS(I,J) = K+1
0063                GO TO 25
0064             23 CONTINUE
0065                PTS(I,J) = X(K)
0066             24 CONTINUE
0067                IPTS(I,J) = K
0068             25 CONTINUE
0069             30 CONTINUE
            C
            C   WRITE OUT CONVERTED POINTS
            C
0070                WRITE(IO,3005)
0071           3005 FORMAT ('ONPTS      POINT 1      POINT 2      POINT 3   ',
           1     '  CONVERTED POINTS'//)
      C
              DO 55 J=1,NB
              NJ = NPTS(J)
           55 WRITE (IO,3006)  NPTS(J), (PTS(I,J),I=1,NJ)
         3006 FORMAT (1X, I4, 3F12.3)
           60 CONTINUE
              WRITE (IO,3007)
              WRITE (IO,3008)  IPTS
         3007 FORMAT('0IPTS')
         3008 FORMAT(2X,3I4,6X,3I4,6X,3I4,6X,3I4,6X,3I4)
      C
              RETURN
              END
```

MODULI

The MODULI routine calculates the effective longitudinal parameters:

INRT	Effective Moment of Inertia
EBAR	Effective Young's Modulus
C	Effective Neutral Axis
E	Steel/Concrete Modulus

from the input data for Steel or Concrete longitudinal data or accepts the effective values as input.

Calculates the upper and lower cross sectional moduli

$$ZBU_{n,j} = \left(\frac{EBAR_{n,j}}{C_{n,j}}\right) \bullet \left(\frac{INRT_{n,j}}{E_{n,j}}\right)$$

$$ZBL_{n,j} = ZBU_{n,j}$$

Calculate the effective lateral parameters

EIL	Effective Young's modulus times the effective moment of inertia
XIL	The effective moment of inertia

from the input lateral beam parameter or the effective lateral beam parameters.

The calculated parameter and intermediate values are printed out if the debug printout is specified, i.e., NDBUG(2) = 1.

No subroutines are called by this routine.

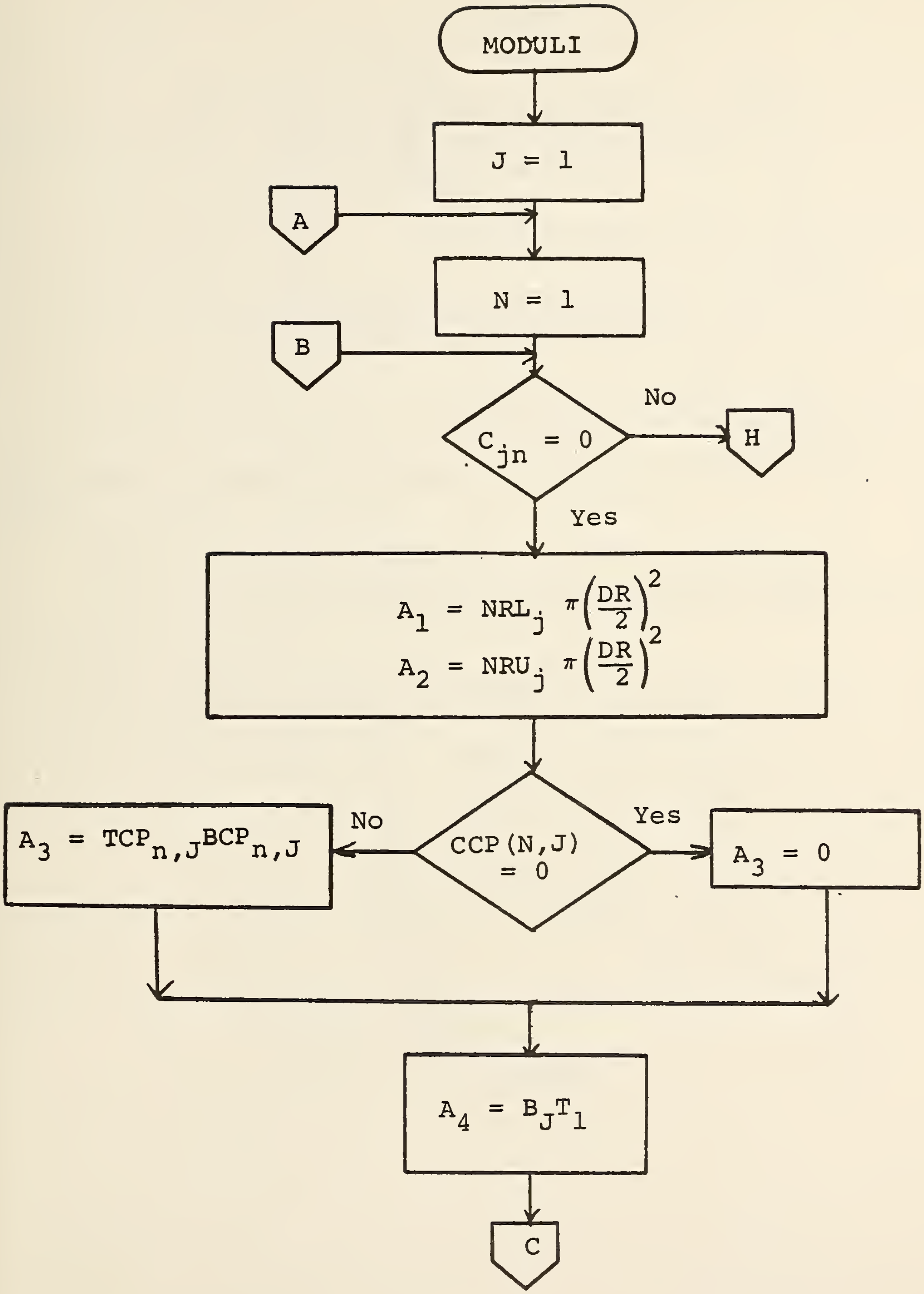

MODULI
J = 1
A
N = 1
B
$C_{jn} = 0$
No
H
Yes
$A_1 = NRL_j \pi\left(\frac{DR}{2}\right)^2$
$A_2 = NRU_j \pi\left(\frac{DR}{2}\right)^2$
$A_3 = TCP_{n,J} BCP_{n,J}$
No
CCP(N,J) = 0
Yes
$A_3 = 0$
$A_4 = B_J T_1$
C

Figure 22. MODULI Program Flow Chart

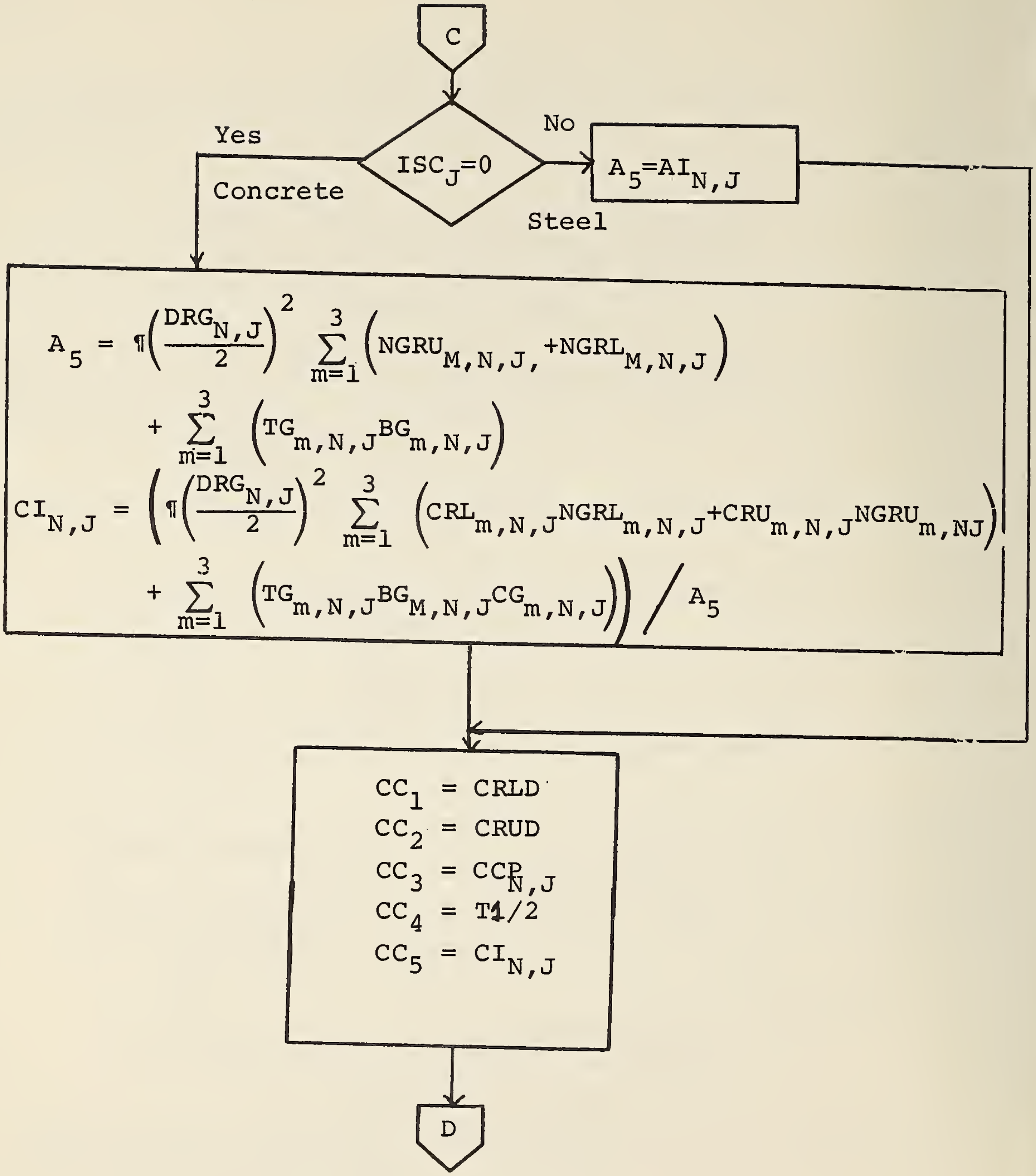

C
$ISC_J=0$
Yes
Concrete
No
Steel
$A_5=AI_{N,J}$
$A_5 = \P\left(\frac{DRG_{N,J}}{2}\right)^2 \sum_{m=1}^{3}\left(NGRU_{M,N,J,} + NGRL_{M,N,J}\right) + \sum_{m=1}^{3}\left(TG_{m,N,J}BG_{m,N,J}\right)$
$CI_{N,J} = \left(\P\left(\frac{DRG_{N,J}}{2}\right)^2 \sum_{m=1}^{3}\left(CRL_{m,N,J}NGRL_{m,N,J} + CRU_{m,N,J}NGRU_{m,NJ}\right) + \sum_{m=1}^{3}\left(TG_{m,N,J}BG_{M,N,J}CG_{m,N,J}\right)\right) \Big/ A_5$
$CC_1 = CRLD$
$CC_2 = CRUD$
$CC_3 = CCP_{N,J}$
$CC_4 = T1/2$
$CC_5 = CI_{N,J}$
D

Figure 22. MODULI Program Flow Chart (Continued)

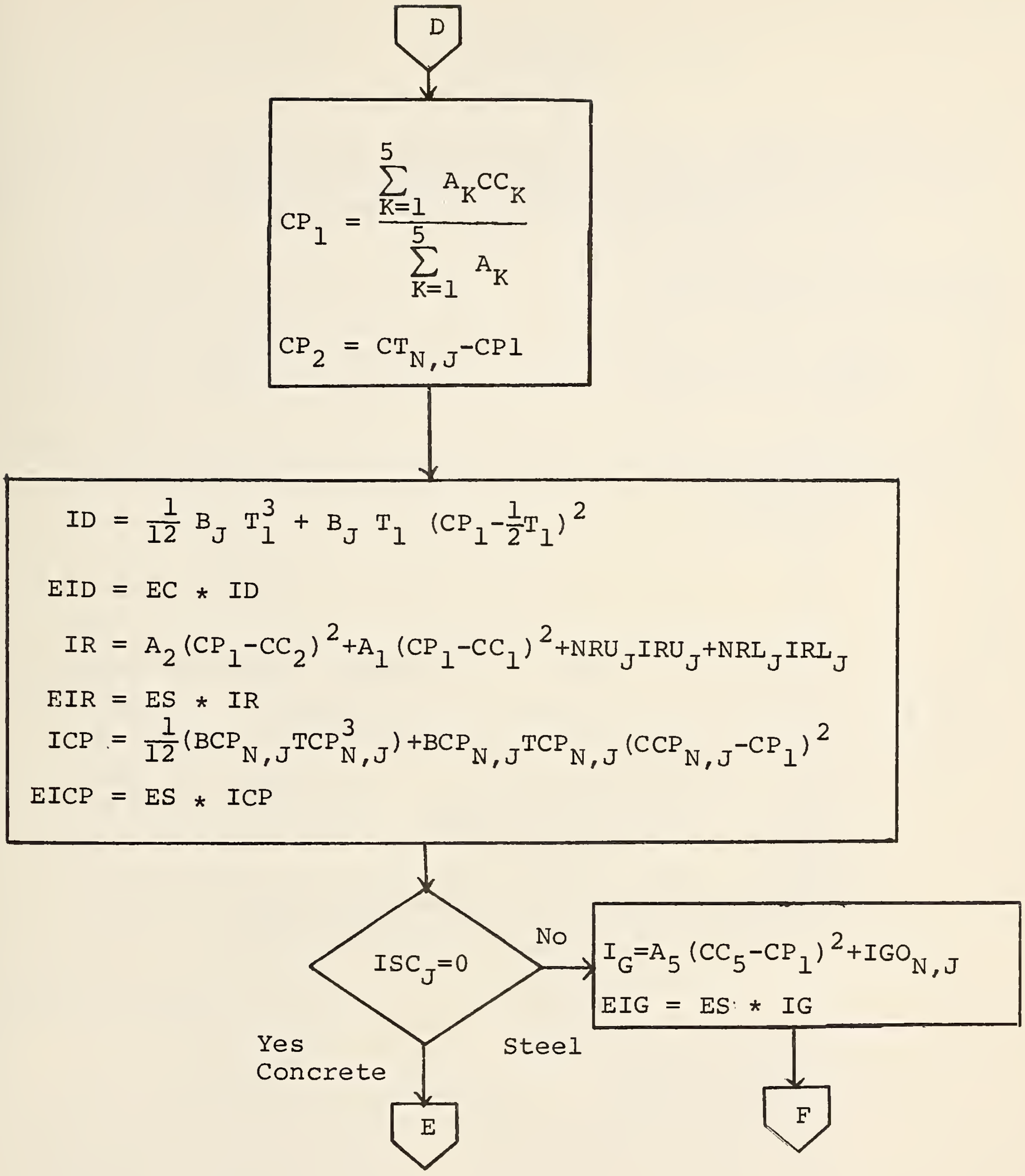
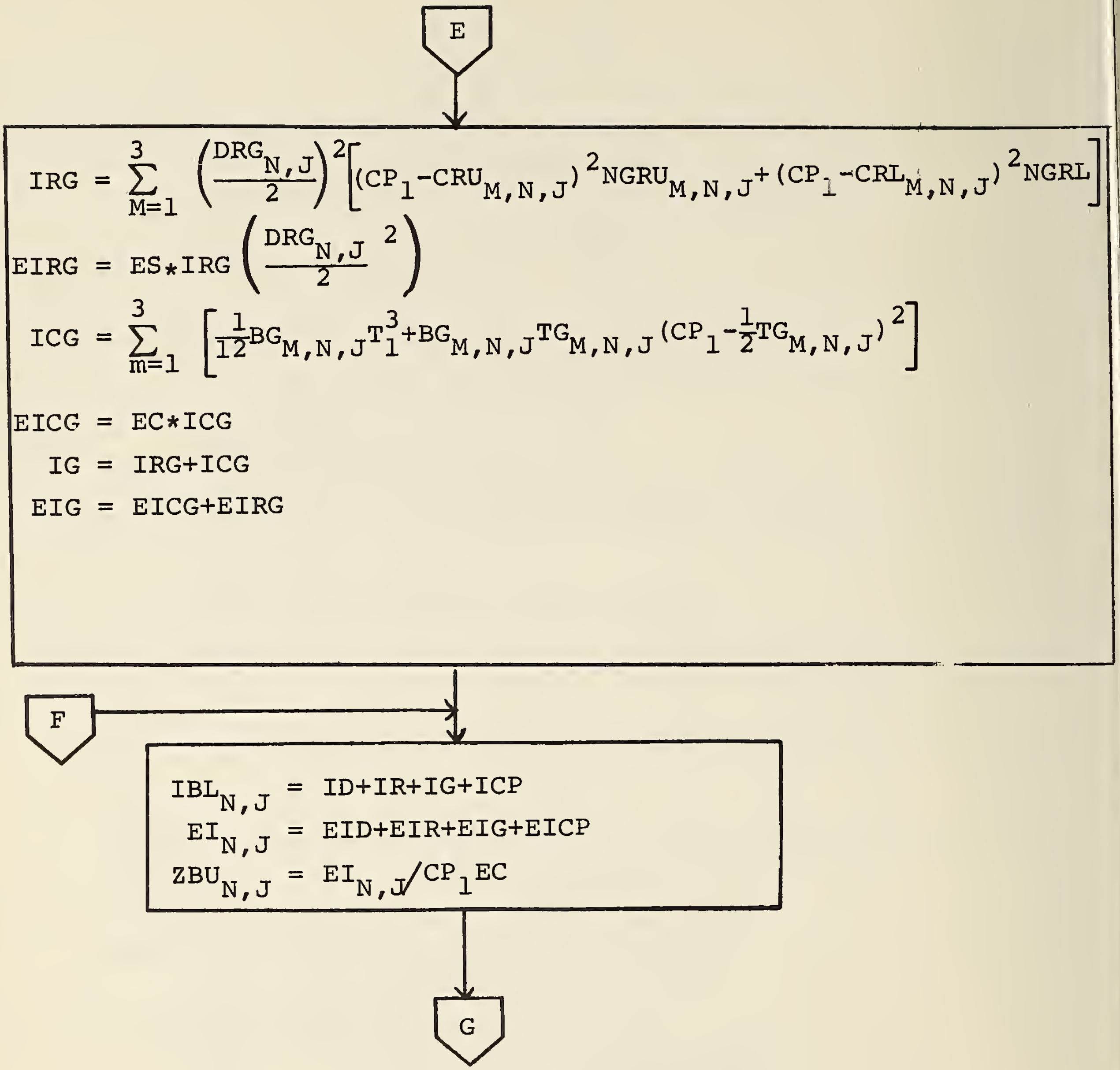

D
$CP_1 = \frac{\sum_{K=1}^{5} A_K CC_K}{\sum_{K=1}^{5} A_K}$
$CP_2 = CT_{N,J} - CP1$
$ID = \frac{1}{12} B_J T_1^3 + B_J T_1 (CP_1 - \frac{1}{2} T_1)^2$
$EID = EC * ID$
$IR = A_2 (CP_1 - CC_2)^2 + A_1 (CP_1 - CC_1)^2 + NRU_J IRU_J + NRL_J IRL_J$
$EIR = ES * IR$
$ICP = \frac{1}{12} (BCP_{N,J} TCP_{N,J}^3) + BCP_{N,J} TCP_{N,J} (CCP_{N,J} - CP_1)^2$
$EICP = ES * ICP$
$ISC_J = 0$
No
Steel
Yes
Concrete
$I_G = A_5 (CC_5 - CP_1)^2 + IGO_{N,J}$
$EIG = ES * IG$
E
F

Figure 22. MODULI Program Flow Chart (Continued)



$$IRG = \sum_{M=1}^{3} \left(\frac{DRG_{N,J}}{2}\right)^{2}\left[(CP_1 - CRU_{M,N,J})^{2} NGRU_{M,N,J} + (CP_1 - CRL_{M,N,J})^{2} NGRL\right]$$

$$EIRG = ES{*}IRG\left(\frac{DRG_{N,J}}{2}{}^{2}\right)$$

$$ICG = \sum_{m=1}^{3}\left[\frac{1}{12}BG_{M,N,J}T_1^{3} + BG_{M,N,J}TG_{M,N,J}\left(CP_1 - \frac{1}{2}TG_{M,N,J}\right)^{2}\right]$$

$$EICG = EC{*}ICG$$

$$IG = IRG+ICG$$

$$EIG = EICG+EIRG$$

$$IBL_{N,J} = ID+IR+IG+ICP$$

$$EI_{N,J} = EID+EIR+EIG+EICP$$

$$ZBU_{N,J} = EI_{N,J}/CP_1 EC$$

Figure 22. MODULI Program Flow Chart (Continued)

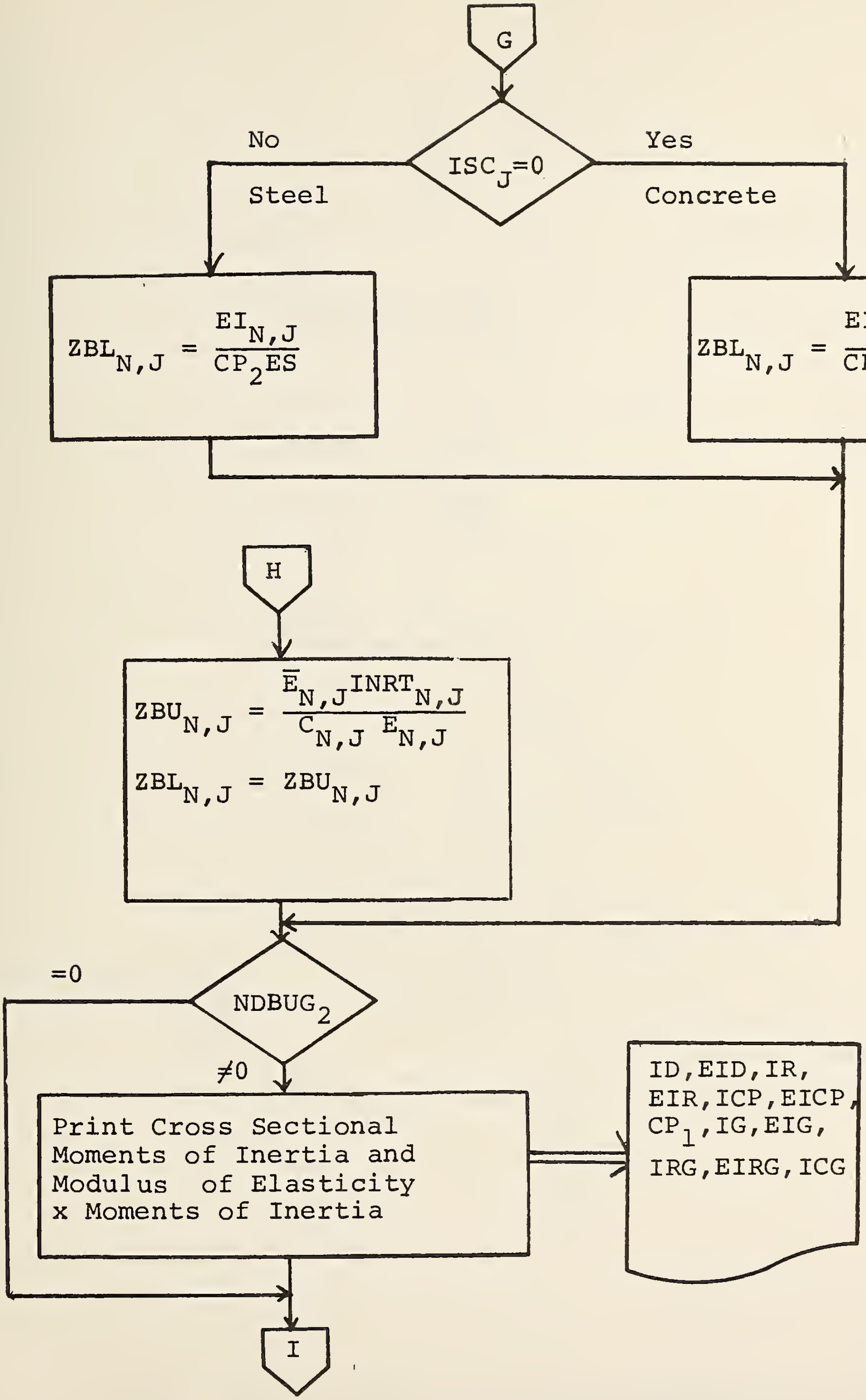

G
ISC_J=0
No
Steel
Yes
Concrete
ZBL_N,J = EI_N,J / CP_2 ES
ZBL_N,J = EI_N,J / CP_2 EC
H
ZBU_N,J = Ē_N,J INRT_N,J / C_N,J E_N,J
ZBL_N,J = ZBU_N,J
NDBUG_2
=0
≠0
Print Cross Sectional Moments of Inertia and Modulus of Elasticity x Moments of Inertia
ID,EID,IR,
EIR,ICP,EICP,
CP_1,IG,EIG,
IRG,EIRG,ICG
I

Figure 22. MODULI Program Flow Chart (Continued)

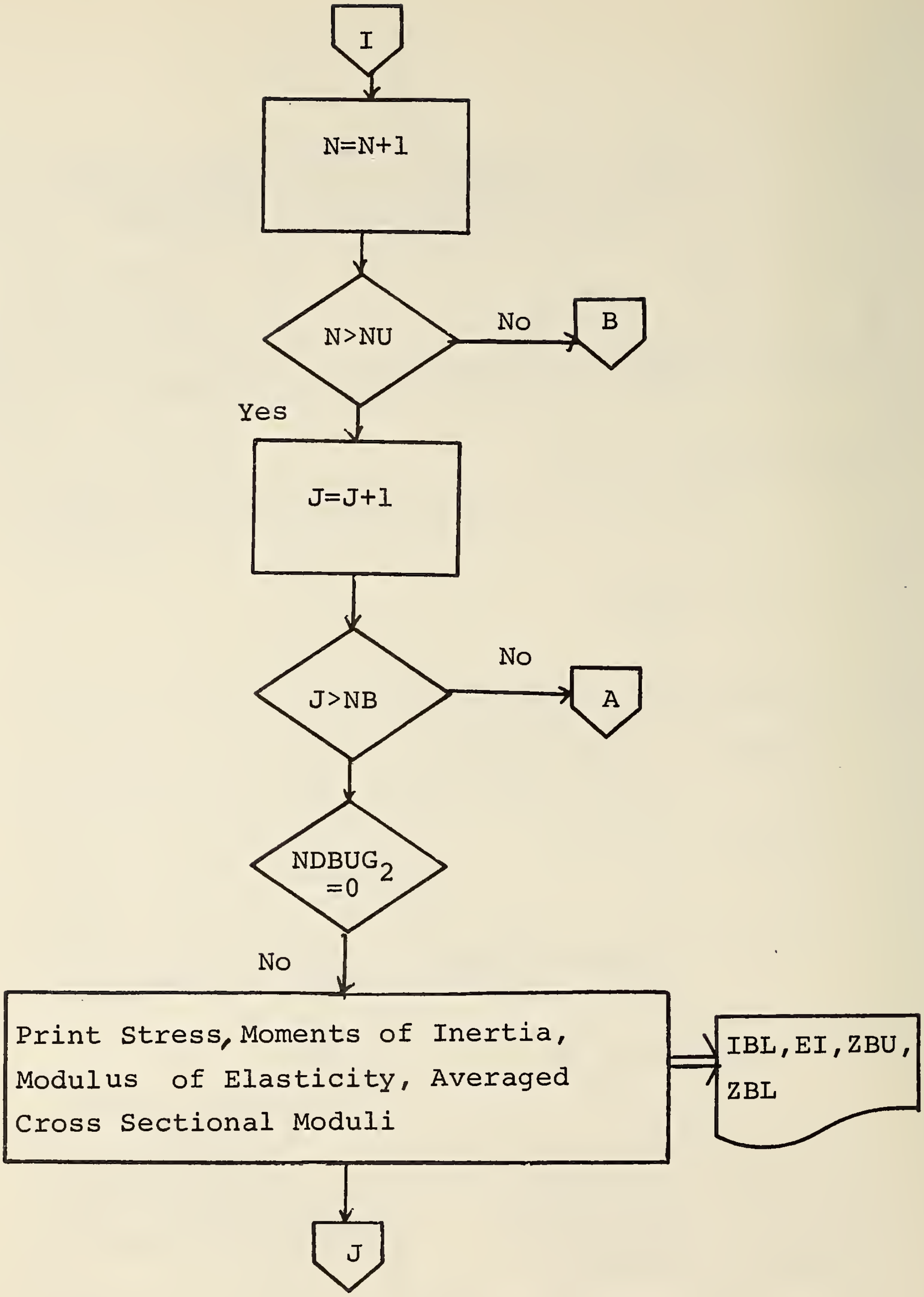

I
N=N+1
N>NU
No
B
Yes
J=J+1
J>NB
No
A
NDBUG2 =0
No
Print Stress, Moments of Inertia, Modulus of Elasticity, Averaged Cross Sectional Moduli
IBL, EI, ZBU, ZBL
J

Figure 22. MODULI Program Flow Chart (Continued)

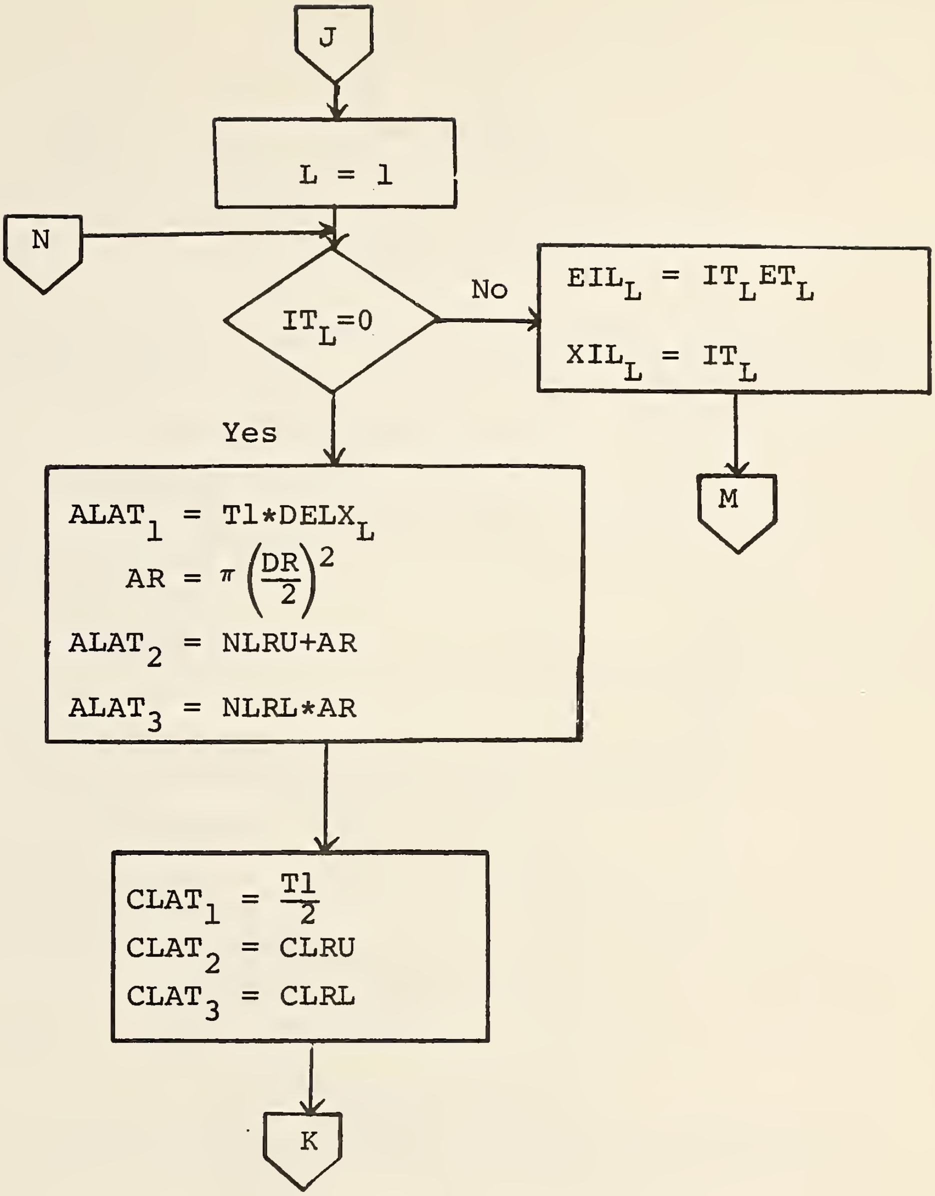


Figure 22. MODULI Program Flow Chart (Continued)

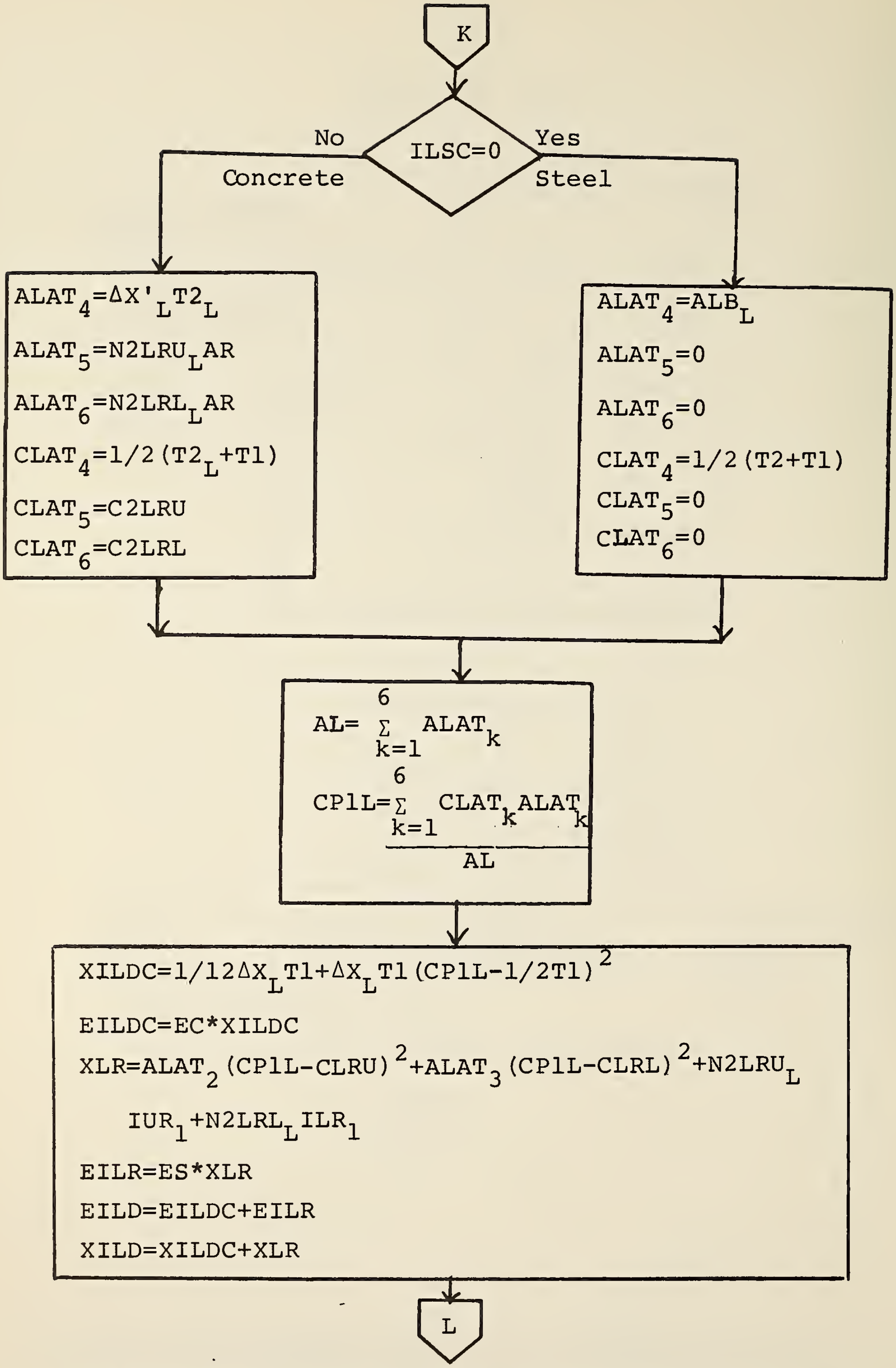


Figure 22. MODULI Program Flow Chart (Continued)

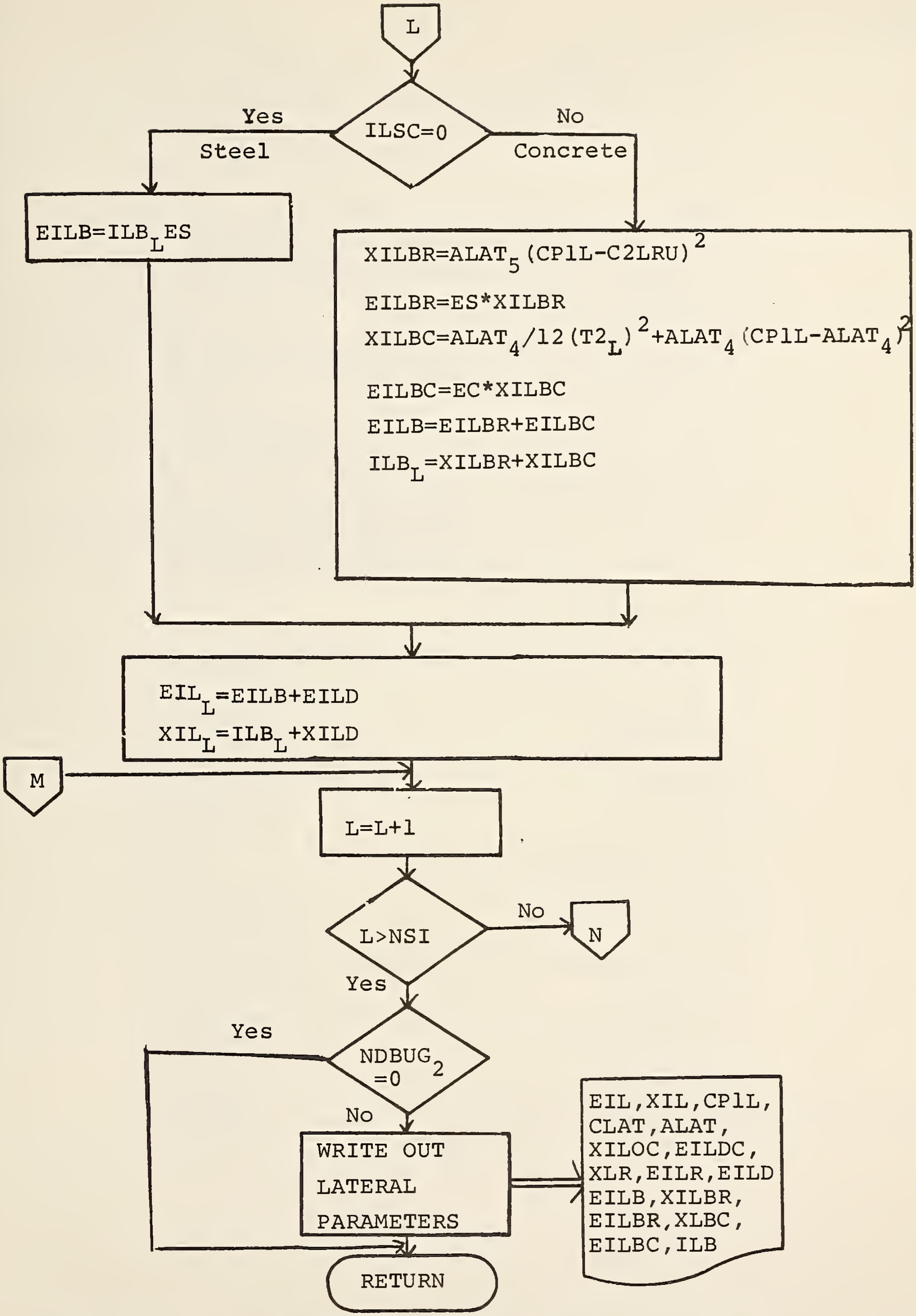

L
ILSC=0
Yes
Steel
No
Concrete
EILB=ILB_L ES
XILBR=ALAT_5 (CP1L-C2LRU)^2
EILBR=ES*XILBR
XILBC=ALAT_4/12 (T2_L)^2+ALAT_4 (CP1L-ALAT_4)^2
EILBC=EC*XILBC
EILB=EILBR+EILBC
ILB_L=XILBR+XILBC
EIL_L=EILB+EILD
XIL_L=ILB_L+XILD
M
L=L+1
L>NSI
No
N
Yes
NDBUG_2 =0
Yes
No
WRITE OUT LATERAL PARAMETERS
EIL,XIL,CP1L, CLAT,ALAT, XILOC,EILDC, XLR,EILR,EILD EILB,XILBR, EILBR,XLBC, EILBC,ILB
RETURN

Figure 22. MODULI Program Flow Chart (Continued)

```
0001                  SUBROUTINE MODULI
            C         THIS SUBROUTINECALCULATES MOMENTS OF INERTIA, NEUTRAL AXES,
            C         EI AND CROSS SECTIONAL MODULI
0002                  REAL NUC,NUS,IUR,ILR,IGO,IRGO,INRT,ILB,IT,ID,ICP,IG,IRG,ICG,IBL,IR
            C
0003                  REAL MD, MDT
0004                  REAL IBL
            C
0005                  COMMON/BLK1/CLRL,CLRU, CLT, CRLD, CRUD, C2LRL, C2LRU, DR, DRT,
                     1   EC, ES, ILSC, NLRL, NLRU, T1, IL, NUC, NUS,
                     2   NRU(10), NRL(10), B(10), ISC(10), ALB(10), N2LRU(10),
                     3   N2LRL(10), T2(10), IT(10), IUR(10), ILR(10),ILB(10),
                     4   DELXP(10), ET(10),
                     5   AI(3,10), CI(3,10), CT(3,10), CCP(3,10), TCP(3,10), BCP(3,10),
                     6   DRG(3,10), C(3,10), IGO(3,10), INRT(3,10), EBAR(3,10),
                     7   BG(3,3,10), CG(3,3,10), TG(3,3,10), NGRU(3,3,10), NGRL(3,3,10),
                     8   CRU(3,3,10), CRL(3,3,10), IRGO(3,3,10)
                     9   ,DUM(1568)
            C
0006                  COMMON /BLK2/ DT,       THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     C
                     1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                     2   NXPTS, NDBUG(10),
                     3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                     4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                     6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                     7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
            C
0007                  COMMON /BLK6/ IBL(3,10)
            C
0008                  DIMENSION CC(5),A(5),ALAT(6),CLAT(6), XIL(10)
            C
0009                  EQUIVALENCE (CPLAT,CP1L)
            C
0010                  DO 30 J=1,10
0011                  DO 29 I=1,3
0012                  IBL(I,J) = 0.0
0013               29 CONTINUE
0014                  XIL(J) = 0.0
0015               30 CONTINUE
            C
0016                  IF (NDBUG(2).EQ.0)  GO TO 40
0017              100 FORMAT (  'OMODULI')
0018                  WRITE (IO,100)
0019               40 CONTINUE
            C
            C       CALCULATE LONGITUDINAL PARAMETERS
0020                  PI=3.141593
            C
0021                  DO 1 J=1,NB
0022                  NU=NCS(J)
            C
0023                  DO 2 N=1,NU
            C
0024                  IF (C(N,J).NE.0) GO TO 13
```

```
      TMP =PI*DR*DR/4.
      A(1)=NRL(J)*TMP
      A(2)=NRU(J)*TMP
C
      IF (CCP(N,J).EQ.0) GO TO 3
      A(3)=TCP(N,J)*BCP(N,J)
      GO TO 4
    3 A(3)=0.
    4 A(4)=B(J)*T1
C
      IF (ISC(J).NE.0) GO TO 6
C
C   CONCRETE
C
      TERM1=0.
      TERM2=0.
      TERM3=0.
      TERM4=0.
      DO 5 M=1,3
      TERM1=TERM1+NGRU(M,N,J)+NGRL(M,N,J)
      TERM2=TERM2+TG(M,N,J)*BG(M,N,J)
      TERM3=TERM3+CRL(M,N,J)*NGRL(M,N,J)+CRU(M,N,J)*NGRU(M,N,J)
    5 TERM4=TERM4+TERM2*CG(M,N,J)
      TMP=PI*DRG(N,J)/2.*DRG(N,J)/2.
      A(5)=TMP*TERM1+TERM2
      CI(N,J)=(TMP*TERM3+TERM4)/A(5)
      GO TO 7
C
C   STEEL
C
    6 A(5)=AI(N,J)
C
    7 CONTINUE
      CC(1)=CRLD
      CC(2)=CRUD
      CC(3)=CCP(N,J)
      CC(4)=T1/2.
      CC(5)=CI(N,J)
      CP1=0.
      DEN=0.
C
      DO 55 K=1,5
      DEN=DEN+A(K)
   55 CP1=CP1+A(K)*CC(K)
C
      CP1=CP1/DEN
      CP2=CT(N,J)-CP1
      ID=B(J)/12.*T1*T1*T1+B(J)*T1*(CP1-T1/2.)**2
      EID=EC*ID
      IR=A(2)*(CP1-CC(2))**2+A(1)*(CP1-CC(1))**2
     1+NRU(J)*IUR(J)+NRL(J)*ILR(J)
      EIR=ES*IR
      ICP=BCP(N,J)/12.*TCP(N,J)**3+BCP(N,J)*TCP(N,J)*(CCP(N,J)-CP1)**2
      EICP=ES*ICP
```

```
            C
0067              IF (ISC(J).EQ.0) GO TO 8
            C
            C   STEEL
            C
0068              IG=A(5)*(CC(5)-CP1)**2+IGO(N,J)
0069              EIG=ES*IG
0070              GO TO 9
            C
            C   CONCRETE
            C
0071            8 TMP=PI*DRG(N,J)/2.*DRG(N,J)/2.
0072              IRG=0.
0073              ICG=0.
0074              DO 10 M=1,3
0075              IRG=IRG+((CP1-CRU(M,N,J))**2*NGRU(M,N,J)+(CP1-CRL(M,N,J))**2*NGRL
                 1(M,N,J)+IRGO(M,N,J))*TMP
0076           10 ICG=ICG+BG(M,N,J)/12.*T1**3+BG(M,N,J)*TG(M,N,J)*(CP1-TG(M,N,J)/2.)
                 1**2
0077              EIRG=ES*IRG
0078              EICG=EC*ICG
0079              IG=IRG+ICG
0080              EIG=EICG+EIRG
            C
0081            9 IBL(N,J)=ID+IR+IG+ICP
0082                EI(N,J)=EID+EIR+EIG+EICP
0083              ZBU(N,J)=  EI(N,J)/(EC*CP1)
            C
0084              IF (ISC(J).NE.0) GO TO 11
            C
            C   CONCRETE
            C
0085              ZBL(N,J) = EI(N,J)/(CP2*EC)
0086              GO TO 12
            C
            C   STEEL
            C
0087           11 ZBL(N,J) = EI(N,J)/(CP2*ES)
0088           12 GO TO 33
0089           13 ZBU(N,J)=EBAR(N,J)/C(N,J)*INRT(N,J)/E(N,J)
0090              ZBL(N,J)=ZBU(N,J)
0091              IBL(N,J) = INRT(N,J)
0092              EI(N,J) = EBAR(N,J)
0093           33 CONTINUE
0094              IF (NDBUG(2).EQ.0)  GO TO 14
0095         1000 FORMAT(1X,10E13.4)
            C
0096              WRITE (IO,1001)
0097         1001 FORMAT ('0A')
0098              WRITE (IO,1000)  A
0099              WRITE (IO,1002)
0100         1002 FORMAT ('0CI')
0101              WRITE (IO,1000)  CI
0102              WRITE (IO,1003)
```

```
0103          1003 FORMAT ('0CC')
0104               WRITE (IO,1000)  CC
0105               WRITE (IO,1004)
0106          1004 FORMAT (                '0         ID           EID            IR',
                  1    '               EIR           ICP           EICP           CP1',
                  2    '               CP2')
0107               WRITE (IO,1000)     ID, EID, IR, EIR, ICP, EICP, CP1, CP2
0108               WRITE (IO,1010)
0109          1010 FORMAT ('0          IG            EIG            IRG           EIRG',
                  1    '               ICG')
0110               WRITE (IO,1000)  IG, EIG, IRG, EIRG, ICG
0111            14 CONTINUE
0112             2 CONTINUE
0113             1 CONTINUE
             C
0114               IF (NDBUG(2).EQ.0)  GO TO 15
0115               WRITE (IO,101)
0116           101  FORMAT ('0   J   N              IBL               EI'
                  1    ,'              ZBU              ZBL')
0117               DO 26 J=1, NB
0118               DO 26 N=1,3
0119               WRITE (IO,102) (J,N,IBL(N,J),  EI(N,J),ZBU(N,J),ZBL(N,J))
0120           102 FORMAT (1X,2I4,4E14.4)
0121            26 CONTINUE
             C
0122            15 CONTINUE
             C
             C     CALCULATE LATERAL PARAMETERS
             C
0123               DO 20 L=1,NSI
0124               IF (IT(L).NE.0.) GO TO 21
0125               ALAT(1)=T1*DELX(L)
0126               AR=DR/2.*DR/2.*PI
0127               ALAT(2)=NLRU*AR
0128               ALAT(3)=NLRL*AR
0129               CLAT(1) =T1/2.
0130               CLAT(2)=CLRU
0131               CLAT(3)=CLRL
             C
0132               IF (ILSC .EQ.0) GO TO 22
             C
             C   CONCRETE
             C
0133               ALAT(4)=DELXP(L)*T2(L)
0134               ALAT(5)=N2LRU(L)*AR
0135               ALAT(6)=N2LRL(L)*AR
0136               CLAT(4)=.5*(T2(L)+T1)
0137               CLAT(5)=C2LRU
0138               CLAT(6)=C2LRL
0139               GO TO 23
             C
             C   STEEL
             C
0140            22 ALAT(4)=ALB(L)
```

```
      ALAT(5)        =0.
              ALAT(6)=0.
      CLAT(4)=.5*(T2(L)+T1)
      CLAT(6)=0.
      CLAT(5)=CLAT(6)
   23 AL=0.
      CP1L=0.
      DO 200 K=1,6
      AL=AL+ALAT(K)
  200 CP1L=CP1L+CLAT(K)*ALAT(K)
      CP1L=CP1L/AL
      XILDC=DELX(L)/12.*T1**3+DELX(L)*T1*(CP1L-T1/2.)**2
      EILDC=EC*XILDC
      XLR=ALAT(2)*(CP1L-CLRU)**2+ALAT(3)*(CP1L-CLRL)**2+N2LRU(L)*IUR(1)+
     1N2LRL(L)*ILR(1)
      EILR=ES*XLR
      EILD=EILDC+EILR
      XILD=XILDC+XLR
      IF (ILSC  .NE. 0)GO TO 24
C
C   STEEL
C
      EILB=ES*ILB(L)
      GO TO 25
C
C   CONCRETE
C
   24 XILBR=ALAT(5)*(CP1L-C2LRU)**2+ALAT(6)*(CP1L-C2LRL)**2
      EILBR=ES*XILBR
      XILBC=ALAT(4)/12.*T2(L)**2+ALAT(4) *(CP1L-CLAT(4))**2
      EILBC=EC*XILBC
      EILB=EILBR+EILBC
      ILB(L)=XILBR+XILBC
C
   25 EIL(L)=EILB+EILD
      XIL(L)=ILB(L)+XILD
C
C     CALCULATE Z -  MODULI FOR LATERAL EFFECT
      GO TO 20
   21 EIL(L)=IT(L)*ET(L)
      XIL(L)=IT(L)
   20 CONTINUE
C
      IF (NDBUG(2).EQ.0)  GO TO 60
      WRITE (IO,103)
  103 FORMAT ('0    L         EIL         XIL')
      WRITE (IO,302) (L,EIL(L),XIL(L),L=1,NSI)
  302 FORMAT (1X,I4, 2E12.3)
      WRITE (IO,1017)
 1017 FORMAT ('0     CP1L         CLAT')
      WRITE (IO,1000)  CP1L, CLAT
      WRITE (IO,1018)
 1018 FORMAT ('OALAT')
      WRITE (IO,1000)  ALAT
```

```
0184                WRITE (IO,1019)
0185           1019 FORMAT ('0      XILDC         EILDC          XLR         EILR',
                   1    '          EILD          EILB         XILBR        EILBR',
                   2    '        XILBC         EILBC')
0186                WRITE (IO,1000)  XILDC, EILDC, XLR, EILR, EILD, EILB, XILBR,
                   1                 EILBR, XILBC, EILBC
0187                WRITE (IO,1030)
0188           1030 FORMAT ('OILB')
0189                WRITE (IO,1000)  ILB
              C
0190             60 CONTINUE
0191                RETURN
0192                END
```

STIFF

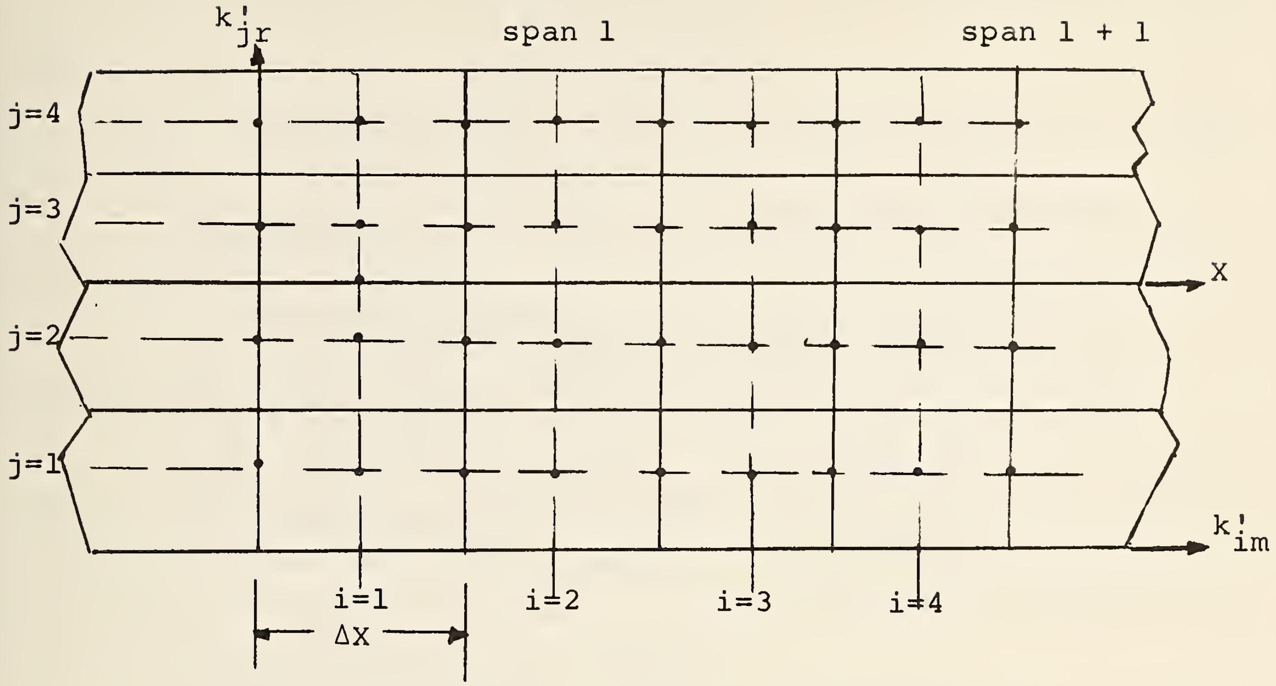
The subroutine STIFF assembles the inverse stiffness matrix, the flexibility matrix, for simple spans.

To assemble the flexibility matrix the following definitions are first required.

1. δ^{j}_{im} is the deflection for a 1# load applied at the i^{th} grid point and on the j^{th} beam, i.e., at the intersection of X_i and Y_{Bjk} (for the L^{th} span). It is evaluated at all longitudinal points for the j^{th} beam of the L^{th} span for $\left(\sum_{p=i}^{L-1} I_{sp}+1\right) \leq m \leq \sum_{p=i}^{L}$ Isp. Hence, there is a vector of these deflection coefficients for each longitudinal beam stiffness relationships.

2. δ'^{i}_{jr} is the deflection of the r^{th} beam for a 1# load applied at the j^{th} beam, and the i^{th}, grid point, i.e., at the intersection of the X_i and the Y_{Bj}. It is evaluated at all lateral points along the i^{th} grid line for $1 \leq r \leq N_B$. Hence, there is a vector of these deflections for each lateral point on each grid line. These reflect the lateral deck, primarily, deflection relationships.

3. The applied loads are assembled as a vector of sequential loads along the grid lines with the grid lines sequenced by their ascending order, e.g.,



Given the deflections at each grid center, for the above example, the deflection vector is

$$\begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \\ \vdots \\ \delta_{16} \end{bmatrix} \quad \text{where } \delta_n = \delta_{(j-1)+i}\text{, i.e., } n=(j-1)i.$$

δ_1 is composed of all the deflections due to loads imposed at points 1,2,3,4,5,9 and 13. Similarly, the virtual load imposed at each grid center are directly related to the composite stiffness-deflection relationships.

This routine calculates the uncoupled effective main beam unit load deflections for a simple span,

$$\delta^{j}_{im} = \frac{(X_{SUPL}-X_{m})(X_{SUPL+1}-X_{i})\left[(X_{SUPL+1}-X_{SUPL})^{2}-(X_{m}-X_{SUPL})^{2}-X_{SUPL+1}-X_{i})^{2}\right]}{6(X_{SUPL+1}-X_{SUPL})EI}$$

where a unit load is applied at i and its effect measured at m.

This routine thus calls STIFFL to compute δ'^{i}_{jr}, the uncoupled unit load deflection of the lateral effective beam. These are then used to calculate the coupled unit load deflections.

If a continuous beam, with more than two supports, is to be evaluated, this subroutine first estimates the deflections in a simple span manner, as described above and, then, adds the unit load discontinuity bending moment contribution to the deflection for each grid, i.e.,

$$\bar{\delta}^{j}_{im} = \delta^{j}_{im} + \frac{X_{SUPL+1}-X_{m}}{EI_{Lj}}\,\Delta X_{L}\left[\hat{M}_{DLij}\left(\frac{X_{m}-X_{SUPL}}{X_{SUPL+1}-X_{SUPL}}\right)\right]$$

where $\bar{\delta}^{j}_{im}$ is used as a replacement of the simple span value δ^{j}_{im} in estimating the final flexibility matrix.

Subroutine STIFFA is then called to assemble the flexibility matrix, that is, the inverse of the stiffness matrix, and to include the coupled effects.

The following subroutines are called:

STIFFL
STIFFA

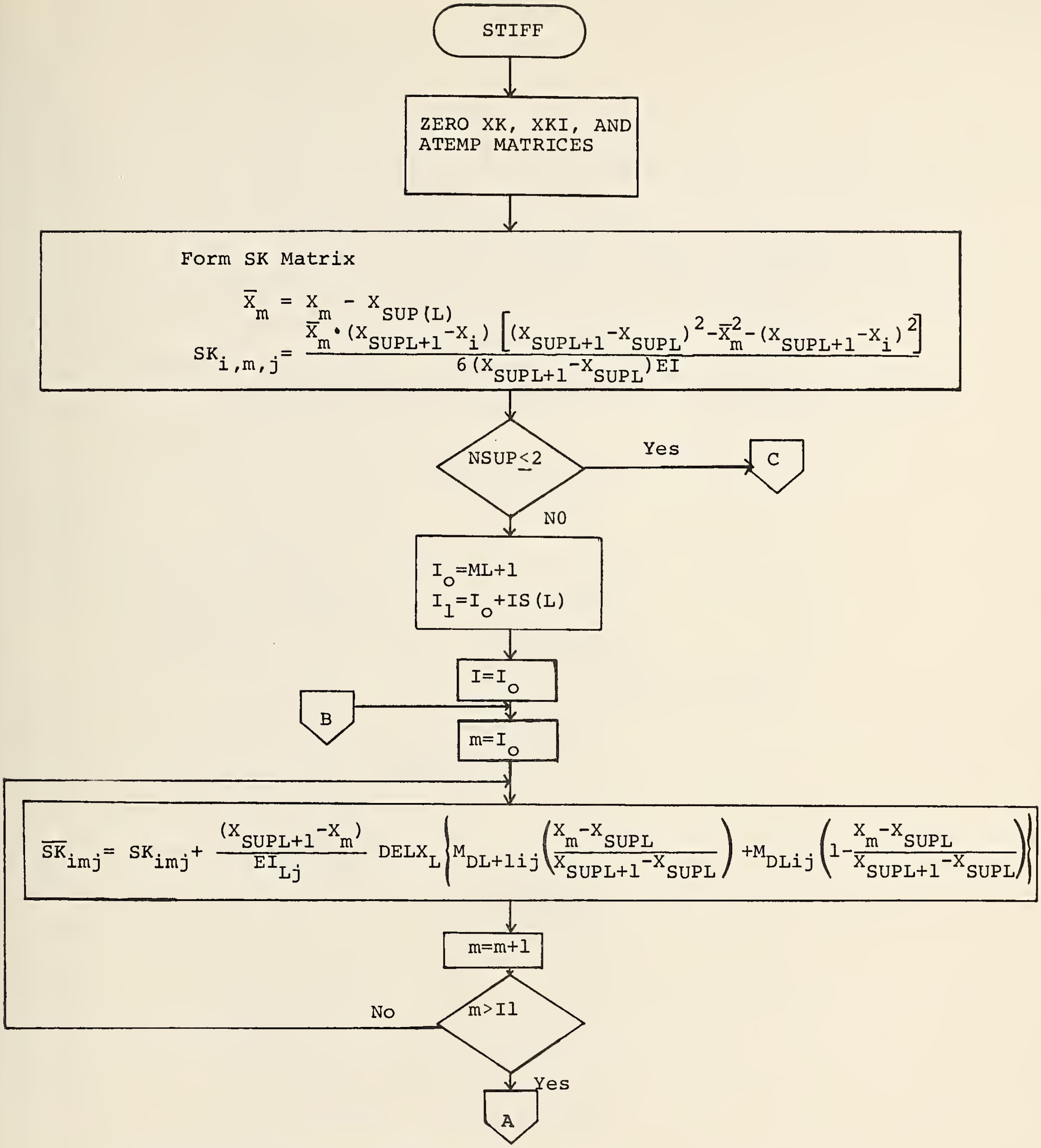

STIFF
ZERO XK, XKI, AND ATEMP MATRICES
Form SK Matrix
$\bar{X}_m = X_m - X_{SUP(L)}$
$SK_{i,m,j} = \frac{\bar{X}_m \cdot (X_{SUPL+1} - X_i) \left[(X_{SUPL+1} - X_{SUPL})^2 - \bar{X}_m^2 - (X_{SUPL+1} - X_i)^2 \right]}{6 (X_{SUPL+1} - X_{SUPL}) EI}$
$NSUP \leq 2$
Yes
C
NO
$I_o = ML+1$
$I_1 = I_o + IS(L)$
$I = I_o$
B
$m = I_o$
$\overline{SK}_{imj} = SK_{imj} + \frac{(X_{SUPL+1} - X_m)}{EI_{Lj}} DELX_L \left\{ M_{DL+1ij} \left(\frac{X_m - X_{SUPL}}{X_{SUPL+1} - X_{SUPL}} \right) + M_{DLij} \left(1 - \frac{X_m - X_{SUPL}}{X_{SUPL+1} - X_{SUPL}} \right) \right\}$
$m = m+1$
No
$m > I1$
Yes
A

Figure 23. STIFF Program Flow Chart

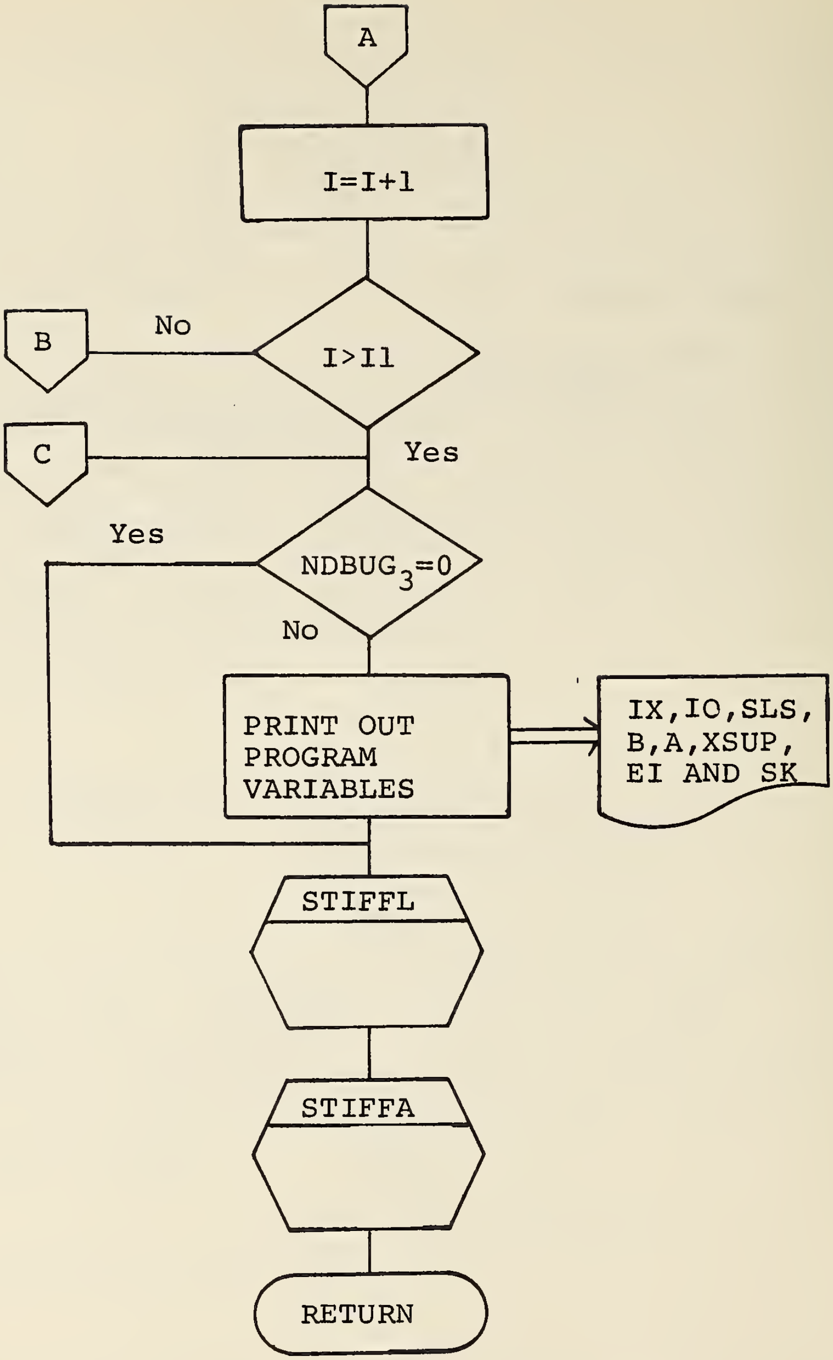


Figure 23. STIFF Program Flow Chart (Continued)

```
0001                  SUBROUTINE STIFF(L)
                C
0002                  INTEGER P, Q, R
0003                  REAL MD, MDT
                C
0004                  COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,
                     1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                     2   NXPTS, NDBUG(10),
                     3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                     4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                     6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                     7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
                C
0005                  COMMON /BLK4/ XK(80,80),XKI(80,80), ATEMP(80,80), DTEMP(80)
                     1          ,MI(20), MB(20), SIGB(20,10)
                C
0006                  DIMENSION SK(20,20,10), SKP(10,10)
0007                  EQUIVALENCE (SK(1,1,1),ATEMP(1,11)), (SKP(1,1),ATEMP(1,51))
                C
                C   ZERO OUT BLK4
                C
0008                  DO 2 I=1,80
0009                  DO 2 J=1,80
0010                  XK(I,J) = 0.0
0011                  XKI(I,J) = 0.0
0012                  ATEMP(I,J) = 0.0
0013                2 CONTINUE
                C
                C   CALCULATE STIFFNESS ELEMENTS
                C
                C
0014                  ML = 0
0015                  LL = L
0016                  DO 100 NN =1,LL
0017              100 ML = ML + IS(NN)
0018                  ML = ML - IS(L)
                C
0019                  IX = IS(L)
0020                  IY = IS(L)/2
0021                  X1  = XSUP(L+1)
0022                  X0 = XSUP(L)
                C
0023                  DO 60 J=1,NB
                C
0024                  IB = 1
0025                  I = IS(L)
0026               10 CONTINUE
0027                  M = I
0028                  MC = IB
0029                  B = X1 - X(I+ML)
0030               20 A = X(M+ML) - X0
0031                  AM = X(M+ML)
                C
0032                  N = 1
```

```
0033               30 IF(AM.LT.XCS(N,J)) GO TO 50
0034                  N = N + 1
0035                  IF(N.LE.NCS(J)) GO TO 30
                C
0036                  STOP 13
                C
0037               50 CONTINUE
0038                  SK(I,M,J) = (B*A*((X1-X0)**2-A**2-B**2))/(6.*(X1-X0)*EI(N,J))
0039                  SK(IB,MC,J) = SK(I,M,J)
0040                  M = M - 1
0041                  MC = MC + 1
0042                  IF (M.GE.1)   GO TO 20
                C
0043                  I = I - 1
0044                  IB = IB + 1
0045                  IF (I.GT.IY)  GO TO 10
                C
0046                  IB = 2
0047                  I = IS(L) - 1
0048               55 M = IX
0049                  MC = 1
0050               57 SK(I,M,J) = SK(M,I,J)
0051                  SK(IB,MC,J) = SK(MC,IB,J)
0052                  M = M - 1
0053                  MC = MC + 1
0054                  IF(M.GT.I)  GO TO 57
0055                  I = I - 1
0056                  IB = IB + 1
0057                  IF(I.GT.IY)  GO TO 55
                C
                C
0058                  IF (NSUP.LE.2) GO TO 80
0059                  I0 = ML + 1
0060                  I1 = I0 + IS(L) - 1
0061                  DO 81 I=I0,I1
0062                  DO 81 M = I0, I1
0063                  SK(I,M,J) = SK(I,M,J)+(X(I)-X0)*DELX(L)/(EEI(L,J)*(X1-X0))*
                     1    ((X1-X(M))*MD(L,I,J) + (X(M)-X0)*MD(L+1,I,J))
0064               81 CONTINUE
0065               80 CONTINUE
0066               60 CONTINUE
                C
0067                  IF (NDBUG(3).EQ.0)  GO TO 87
0068                  WRITE (IO,1010)
0069             1010 FORMAT ('OXSUP ')
0070                  WRITE (IO,1011)  XSUP
0071             1011 FORMAT(1X,10E13.4)
                C
0072                  WRITE  (IO, 1025)
0073             1025 FORMAT('OEI')
0074                  WRITE (IO,1011) EI
                C
0075                  ISL = IS(L)
0076                  WRITE (IO,1019)
```

```
0077          1019 FORMAT('0    M    J    SK')
0078               DO 69 J=1,NB
0079               DO 70 M=1,ISL
0080               WRITE (IO,1020) M,J,(SK(I,M,J),I=1,ISL)
0081          1020 FORMAT(1X,2I4,10E12.3/1X,8X,10E12.3)
0082            70 CONTINUE
0083            69 CONTINUE
0084            87 CONTINUE
0085               CALL STIFFL (L)
0086               CALL STIFFA (L)
              C
0087               RETURN
0088               END
```

STIFFL

The subroutine calculates the lateral unit load deflections δ' as

$$\delta'_{jr} = \frac{(Y_{SUP}-Y_{Bj})\ Y_{Bj}\,(Y_{SUP}^{2} - Y_{Bj}^{2} - (Y_{SUP} - Y_{Bj})^{2})}{6Y_{SUP}E_{IL}}$$

No subroutines are called by this routine.

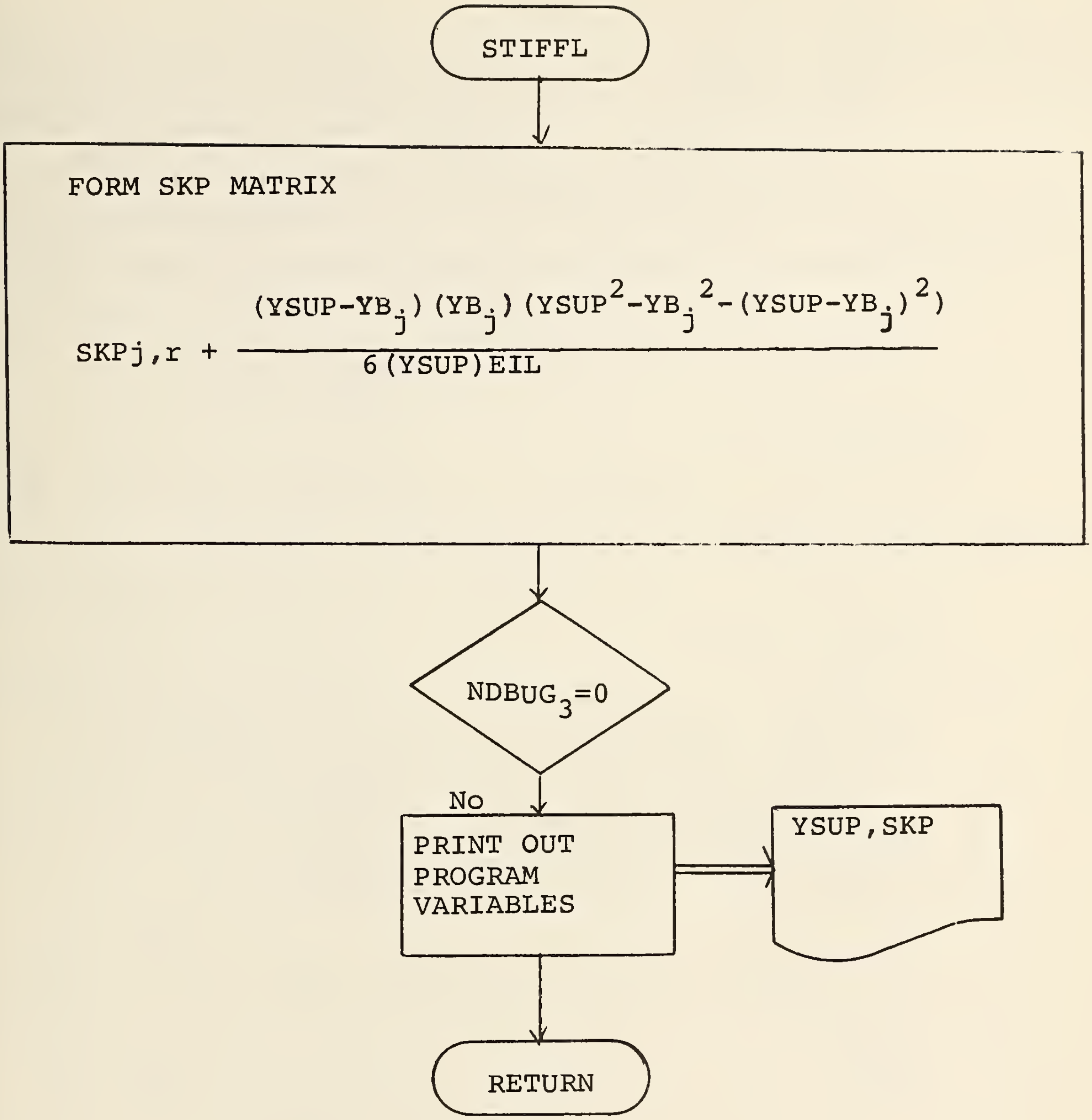

STIFFL
FORM SKP MATRIX
$SKPj,r + \frac{(YSUP-YB_j)(YB_j)(YSUP^2-YB_j{}^2-(YSUP-YB_j)^2)}{6(YSUP)EIL}$
$NDBUG_3=0$
No
PRINT OUT PROGRAM VARIABLES
YSUP, SKP
RETURN

Figure 24. STIFFL Program Flow Chart

```
              SUBROUTINE STIFFL (L)
      C
              REAL MD, MDT
      C
              COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     (
             1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
             2   NXPTS, NDBUG(10),
             3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
             4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
             6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
             7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
      C
              COMMON /BLK4/ XK(80,80),XKI(80,80), ATEMP(80,80), DTEMP(80)
             1              ,MI(20), MB(20), SIGB(20,10)
      C
              DIMENSION SK(20,20,10), SKP(10,10)
              EQUIVALENCE (SK(1,1,1),ATEMP(1,1)), (SKP(1,1),ATEMP(1,51))
      C
              IF (NDBUG(3).EQ.0)  GO TO 77
               WRITE (IO,1010)
         1010 FORMAT ('0        YSUP    J          B   JR          A       SKP')
           77 CONTINUE
      C
              YSUP = 2*YBCL
              J = NB
           80 CONTINUE
      C
              B = YSUP - YB(J)
              JR = J
           90 CONTINUE
      C
              A = YB(JR)
              SKP(J,JR)   = (B*A*(YSUP**2-A**2-B**2))/(6.*YSUP*EIL(L))
              SKP(JR,J)   = SKP(J,JR)
      C
              IF (NDBUG(3).EQ.0)  GO TO 87
              WRITE (IO,1012) YSUP, J, B, JR, A, SKP(J,JR)
         1012 FORMAT (1X,E12.3, I4, E12.3,I4,2E12.3)
           87 CONTINUE
      C
              JR = JR - 1
              IF (JR.NE.0)  GO TO 90
              J = J - 1
              IF (J.NE.0)  GO TO 80
      C
              RETURN
              END
```

STIFFA

This subroutine includes the coupled effect, as described below, and assembles the flexibility matrix. The coupling is accomplished at the same time as the assembly.

For the main diagonal deflection coefficients, that is, at the point of coupling, it can be shown that the total deflection at the point of coupling, δ'_{ii}, is

$$\delta'_{ii} = \frac{\delta'_{ii}\ \delta_{ii}}{\delta'_{ii}+\delta_{ii}}$$

where δ'_{ii} is the deflection of the diaphram at ii due to a 1# load at ii and δ_{ii} is the deflection of the effective main beam at ii due to a 1# load at ii.

The proof is as follows:

given:

$$\bar{\delta}_{ii} = \frac{f'_{ii}}{6EI'}\ ,\quad A, = f'_{ii}\ C' = f'_{ii}\ \delta_{ii}$$

$$\bar{\delta}_{ii} = \frac{f_{ii}}{6EI}\qquad A_2 = f_{ii}\ C = f_{ii}\ \delta_{ii}$$

and

$$f'_{ii} + f_{ii} = 1\#$$

then

$$f'_{ii} = 1 - f_{ii}.$$

substituting

$$\bar{\delta}_{ii} = (1 - f_{ii})\delta_{ii}$$

and

$$f_{ii}\delta_{ii} = (1 - f_{ii})\delta'_{ii}$$

$$f_{ii}\delta_{ii} = {}_{ii} - f_{ii}\delta'_{ii}$$

$$f_{ii} = \frac{\delta'_{ii}}{\delta_{ii}+\delta'_{ii}}$$

Substituting for f_{ii}

$$\bar{\delta}_{ii} = \frac{\delta'_{ii}\delta_{ii}}{\delta_{ii}+\delta'_{ii}}$$

To analyze the physical implications we change variables to the stiffness coefficients, as if each beam were uncoupled, that is,

$$\frac{1}{\bar{k}_{ii}} = \frac{1/k'_{ii}\ 1/k_{ii}}{1/k_{ii}+1/k'_{ii}}$$

Simplifying:

$$\bar{k}_{ii} = k_{ii} + k'_{ii} .$$

This reflects a parallel stiffness effect, analogous to two parallel springs of different strengths, which is exactly the case at the point of coupling.

Hence, the total unit load deflection at the point of coupling can be determined from the two uncoupled unit load deflections.

For a far coupled point in the lateral direction, the deflection at a point ij due to a unit load at ii can be determined by extending the above analysis. Since f' is that fraction of the unit load being applied to the diaphram, then

$$\bar{\delta}_{ij} = f'_{ii}\delta'_{ij}$$

where δ'_{ij} is the uncoupled deflection at ij due to a unit load at ii, and $\bar{\delta}_{ij}$ is the resulting deflection at ij due to a coupled effect at ii and a unit load applied at ii.

Since

$$f'_{ii} = \frac{\delta_{ii}}{\delta_{ii}+\delta'_{ii}}$$

$$\bar{\delta}_{ij} = \frac{\delta'_{ij}\ \delta_{ii}}{\delta_{ii}+\delta'_{ii}}$$, and again

the coupled deflection can be determined from the unit load uncoupled deflections.

To analyze the physical implications we substitute the uncoupled stiffness coefficients,

$$\frac{1}{\bar{k}_{ij}} = \frac{1/k'_{ij}\ 1/k_{ii}}{1/k_{ii}+1/k'_{ii}}$$

Simplifying

$$\bar{k}_{ij} = \frac{k'_{ij}}{k'_{ii}}\ (k'_{ii}+k_{ii}).$$

This implies again, a parallel effect but modified by a proportionality factor determined from the relationship of the uncoupled stiffness coefficients at the point of coupling and at the far point.

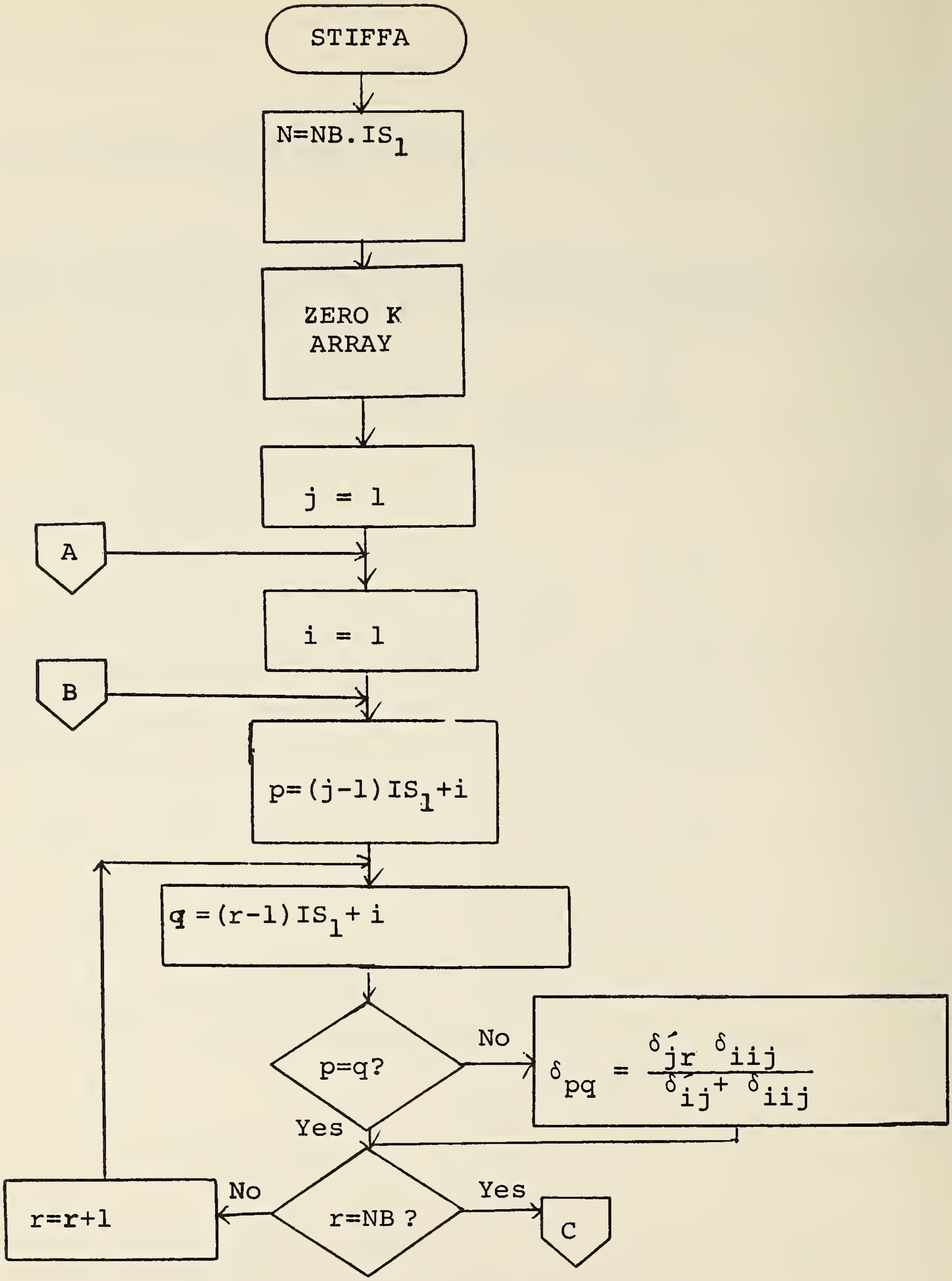


Figure 25. STIFFA Program Flow Chart

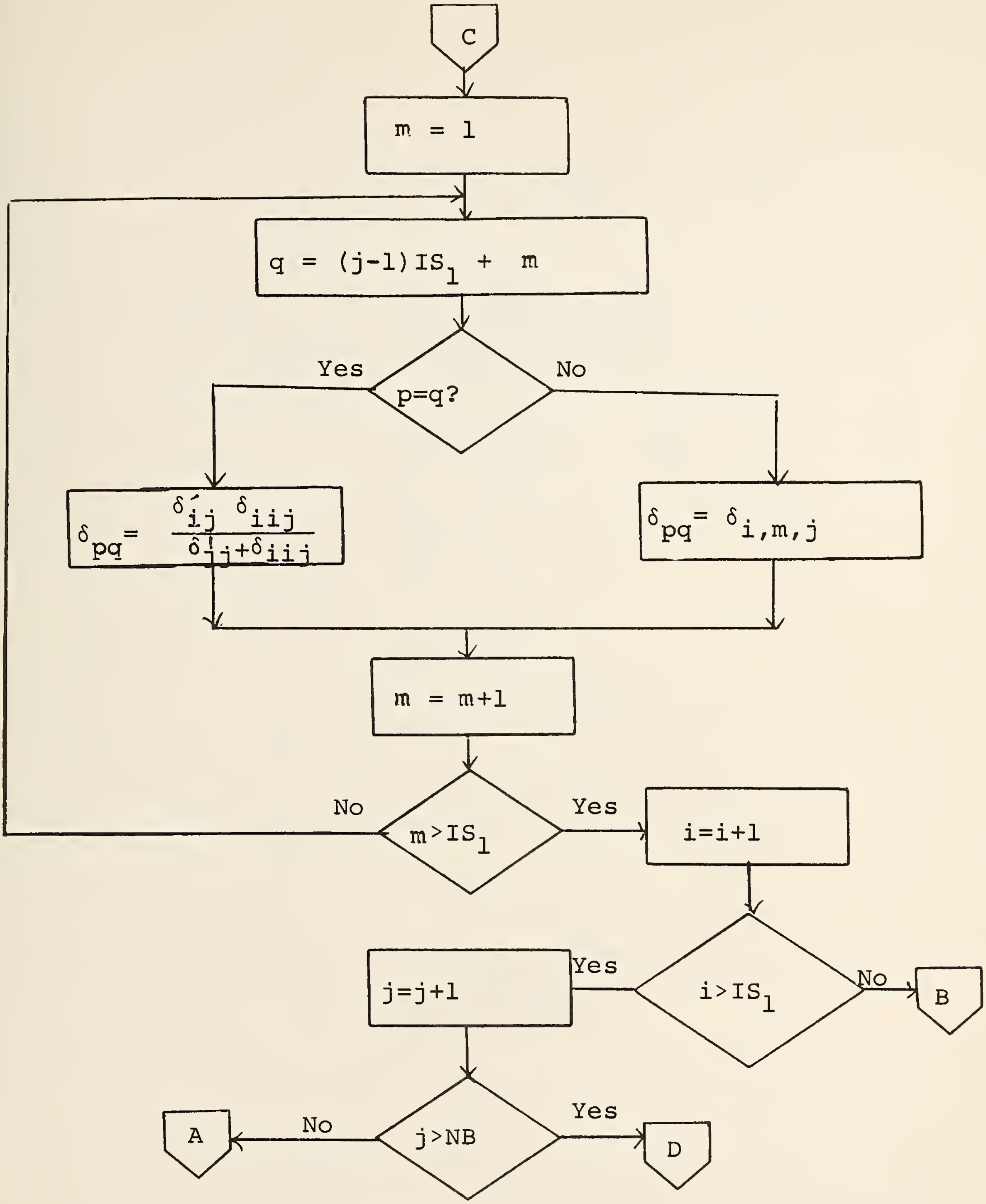


Figure 25. STIFFA Program Flow Chart (Continued)

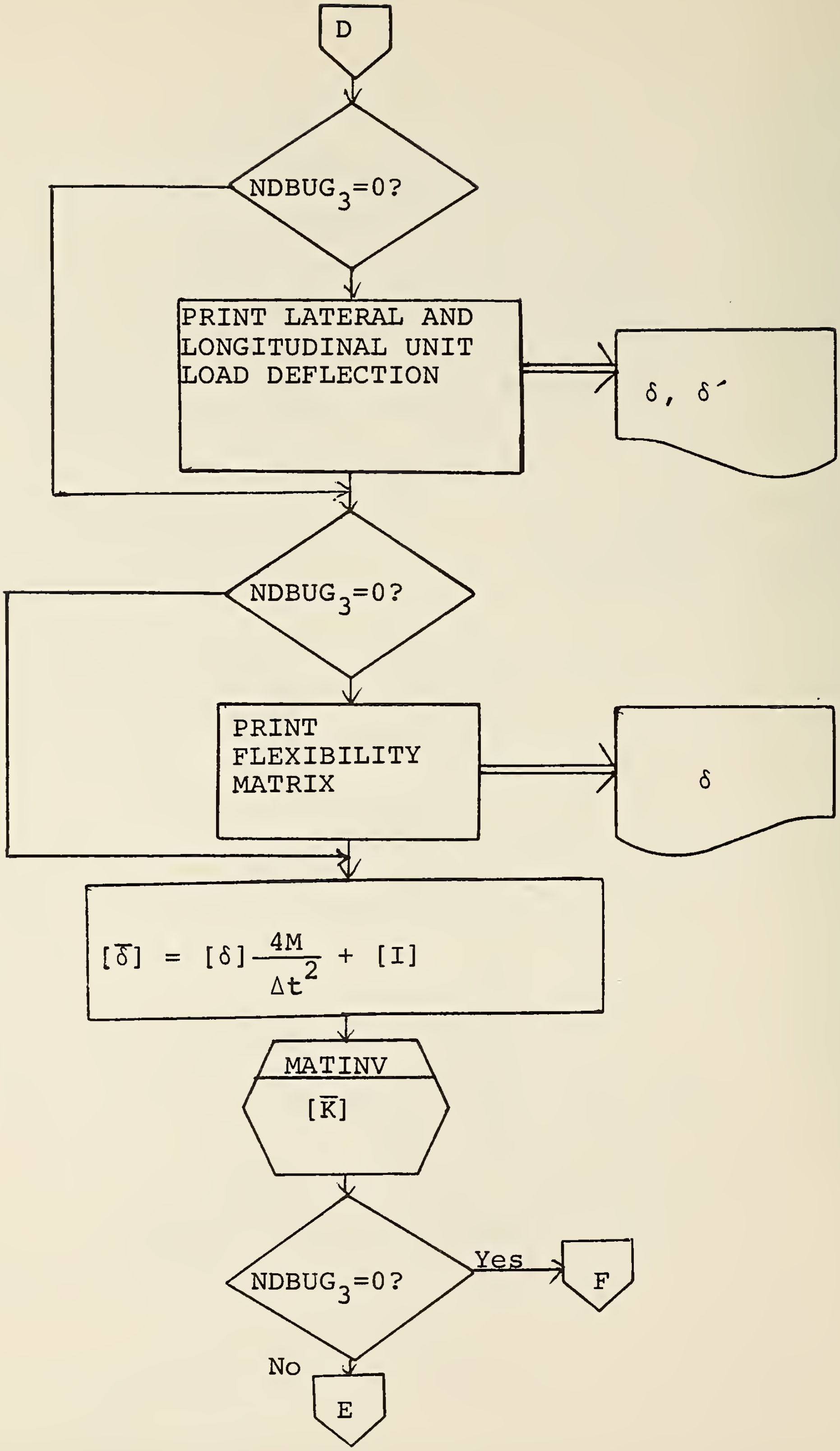


Figure 25. STIFFA Program Flow Chart (Continued)

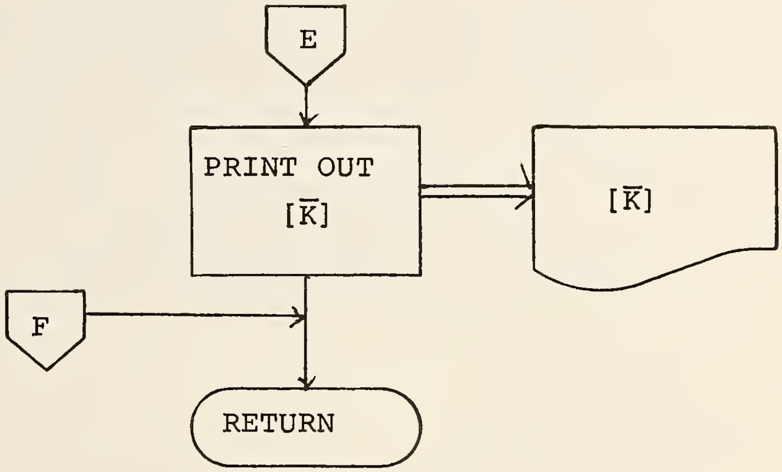


Figure 25. STIFFA Program Flow Chart (Continued)

```
0001                  SUBROUTINE STIFFA (L)
                C
                C   ASSEMBLE STIFFNESS MATRIX
                C
0002                  INTEGER P, Q, R
0003                  REAL MD, MDT
                C
0004                  COMMON /BLK1/ XKBAR(80,80)
                C
0005                  COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     (
                     1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                     2   NXPTS, NDBUG(10),
                     3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                     4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                     6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                     7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
                C
0006                  COMMON /BLK4/ XK(80,80),XKI(80,80), ATEMP(80,80), DTEMP(80)
                     1         ,MI(20), MB(20), SIGB(20,10)
                C
0007                  DIMENSION SK(20,20,10), SKP(10,10)
0008                  EQUIVALENCE (SK(1,1,1),ATEMP(1,1)), (SKP(1,1),ATEMP(1,51))
                C
0009                  N = NB*IS(L)
                C
0010                  DO 100 P=1,N
0011                  DO 100 Q=1,N
0012              100 XK(P,Q) = 0.0
                C
0013                  DO 400 J=1,NB
0014                  ISL = IS(L)
0015                  DO 400 I=1,ISL
0016                  P = (J-1)*IS(L) + I
                C
0017                  DO 200 R=1,NB
0018                  Q=(R-1)*IS(L) + I
0019                  IF (P.EQ.Q)  GO TO 200
0020                  XKI(P,Q) =  (SK(I,I,J)*SKP(J,R))/(SK(I,I,J) + SKP(J,J))
0021              200 CONTINUE
                C
0022                  DO 320 M=1,ISL
0023                  Q = (J-1)*IS(L) + M
0024                  IF (P.EQ.Q)  GO TO 300
0025                  XKI(P,Q) =  SK(I,M,J)
                C
0026                  GO TO  320
0027              300 XKI(P,Q) =   (SK(I,I,J) * SKP (J,J))/(SK(I,I,J) + SKP(J,J))
                C
0028              320 CONTINUE
0029              400 CONTINUE
                C
0030                  IF(NDBUG(3).NE.1)  GO TO 450
                C
0031                  WRITE (IO,1400) ISL,ISL,NB
```

```
 1400 FORMAT ('1SK ',3I6)
C
      DO 420 I=1,NB
      DO 410 J=1,ISL
  410 WRITE (IO,1320)   (SK(K,J,I),K=1,ISL)
      WRITE (IO,1401)
 1401 FORMAT ('0')
  420 CONTINUE
C
      WRITE (IO,1405) NB, NB
 1405 FORMAT ('0SKP',3I6)
      DO 430 J=1,NB
  430 WRITE (IO,1320)  (SKP(K,J),   K=1,NB)
C
  450 CONTINUE
C
      MAXIT = 3
      IERR = 0
C
C    WRITE OUT THE STIFFNESS MATRIX FOR EACH SPAN
C
      IF (NDBUG(3).NE.1)  GO TO 350
      WRITE(IO,1139)
 1139 FORMAT('1')
C
      WRITE(IO,1139)
      DO 340 P=1,N
      WRITE (IO,1140) P
 1140 FORMAT (1X,'XKI',I4)
      WRITE (IO,1320) (XKI(P,Q),Q=1,N)
 1320 FORMAT (1X,10E13.4)
  340 CONTINUE
C
C
  350 CONTINUE
C
C
  800 CONTINUE
      DO 850 I=1,N
      DO 850 J=1,N
      XKBAR(I,J) = XKI(I,J) * ((4. * DELW(L))/DT**2)
      IF(I.EQ.J)  XKBAR(I,J) = XKBAR(I,J) + 1.0
  850 CONTINUE
      CALL MATINV (80,N,XKBAR,XK,ATEMP, DTEMP, MAXIT, IERR)
C
C
C    WRITE OUT THE STIFFNESS MATRIX MODIFIED
C
      IF (NDBUG(3).NE.1)  GO TO 860
      WRITE(IO,1139)
      DO 830 P=1,N
      WRITE(IO,1830) P
 1830 FORMAT(1X,'XK',I4,'  MODIFIED')
      WRITE (IO,1320)  (XK(P,Q),Q=1,N)
```

```
0070             830 CONTINUE
0071             860 CONTINUE
               C
0072                 RETURN
0073                 END
```

The basis of the continuous main beam (girder) solution utilized by this program is the division of a bridge into free bodies that appear to be simple spans. In order to reconstitute the bridge as one body continuity must be re-established after generating the simple span stresses. This is accomplished by developing a set of unit load discontinuity moment vectors in the initialization portion of the program and operating in these vectors during the dynamic calculations with the actual live load to determine the actual discontinuity moments. The contribution of the actual discontinuity moments to the individual grid (element) dynamic stress is then calculated and added to the simple span stress. By using this approach, i.e., initialization of unit load discontinuity moment vector, calculational time is minimized. This is especially significant because of the inversion of a matrix in the calculations.

This subroutine calculates, in the initialization portion of the program, the unit load discontinuity vector for each grid (element) as if a unit load were applied on that grid. It further generates a set of these vectors for each effective beam, i.e.,

$$\left[M_D\right]_{ij} = -6 \left[A\right]_j^{-1} \left[B\right]_{ij}$$

where $\left[A\right]$ and $\left[B\right]$ were defined in the basic reports.

Since it must calculate the value of EI for each effective beam and span in order to calculate the unit load discontinuity moments, it also

1. Interpolates for EI as a function of position of change in the effective beam's cross section for each span, e.g., in the case of a bottom lever.

2. Calculates the cross sectional moduli, ZCU_1 and ZCU_1, for each span and beam.

No subroutines are called by this routine.

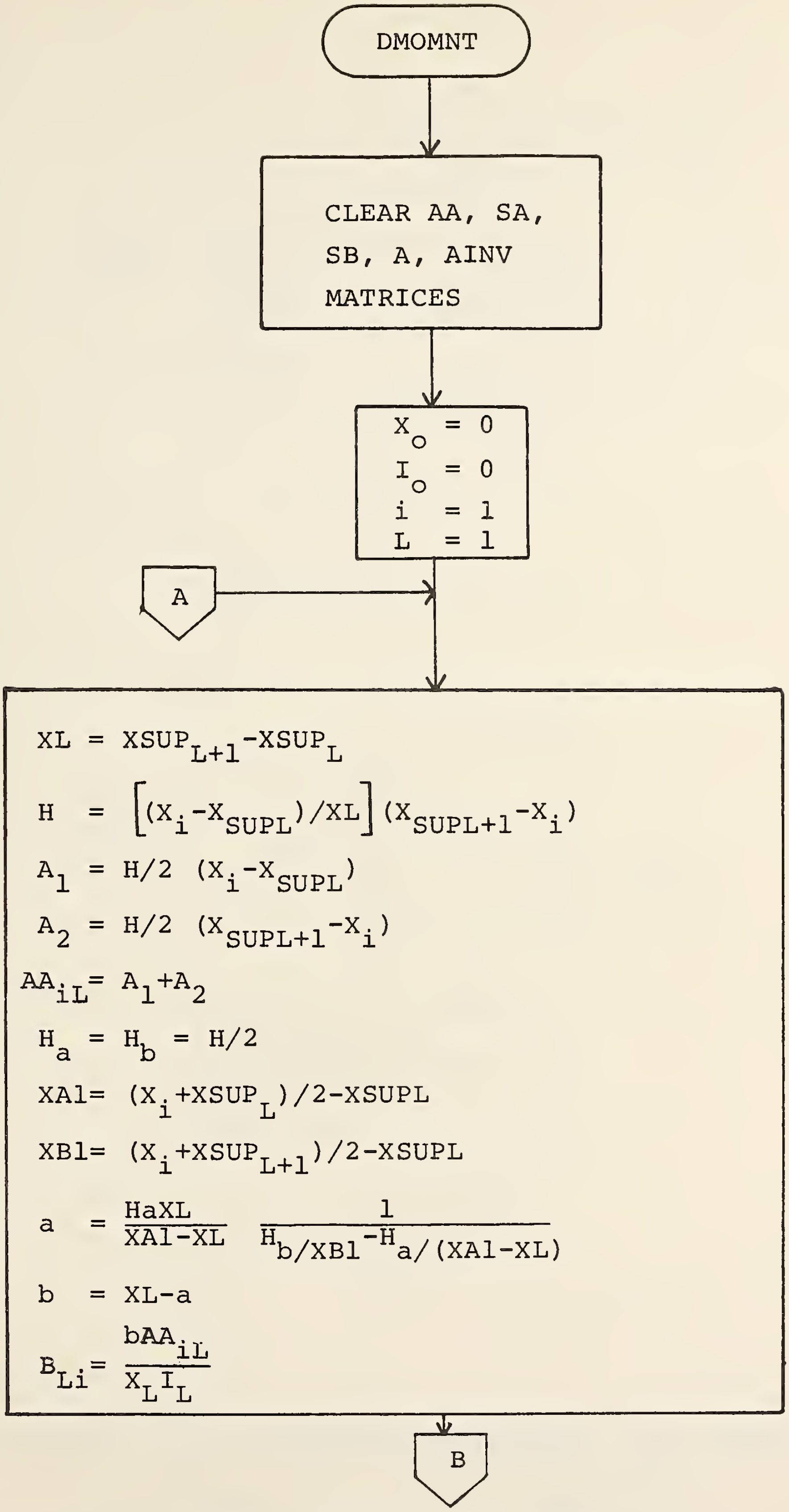

DMOMNT
CLEAR AA, SA, SB, A, AINV MATRICES
$X_o = 0$
$I_o = 0$
$i = 1$
$L = 1$
A
$XL = XSUP_{L+1} - XSUP_L$
$H = \left[(X_i - X_{SUPL})/XL\right](X_{SUPL+1} - X_i)$
$A_1 = H/2\ (X_i - X_{SUPL})$
$A_2 = H/2\ (X_{SUPL+1} - X_i)$
$AA_{iL} = A_1 + A_2$
$H_a = H_b = H/2$
$XA1 = (X_i + XSUP_L)/2 - XSUPL$
$XB1 = (X_i + XSUP_{L+1})/2 - XSUPL$
$a = \frac{HaXL}{XA1-XL}\ \frac{1}{H_{b/XB1} - H_{a/(XA1-XL)}}$
$b = XL - a$
$B_{Li} = \frac{bAA_{iL}}{X_L I_L}$
B

Figure 26. DMOMNT Program Flow Chart

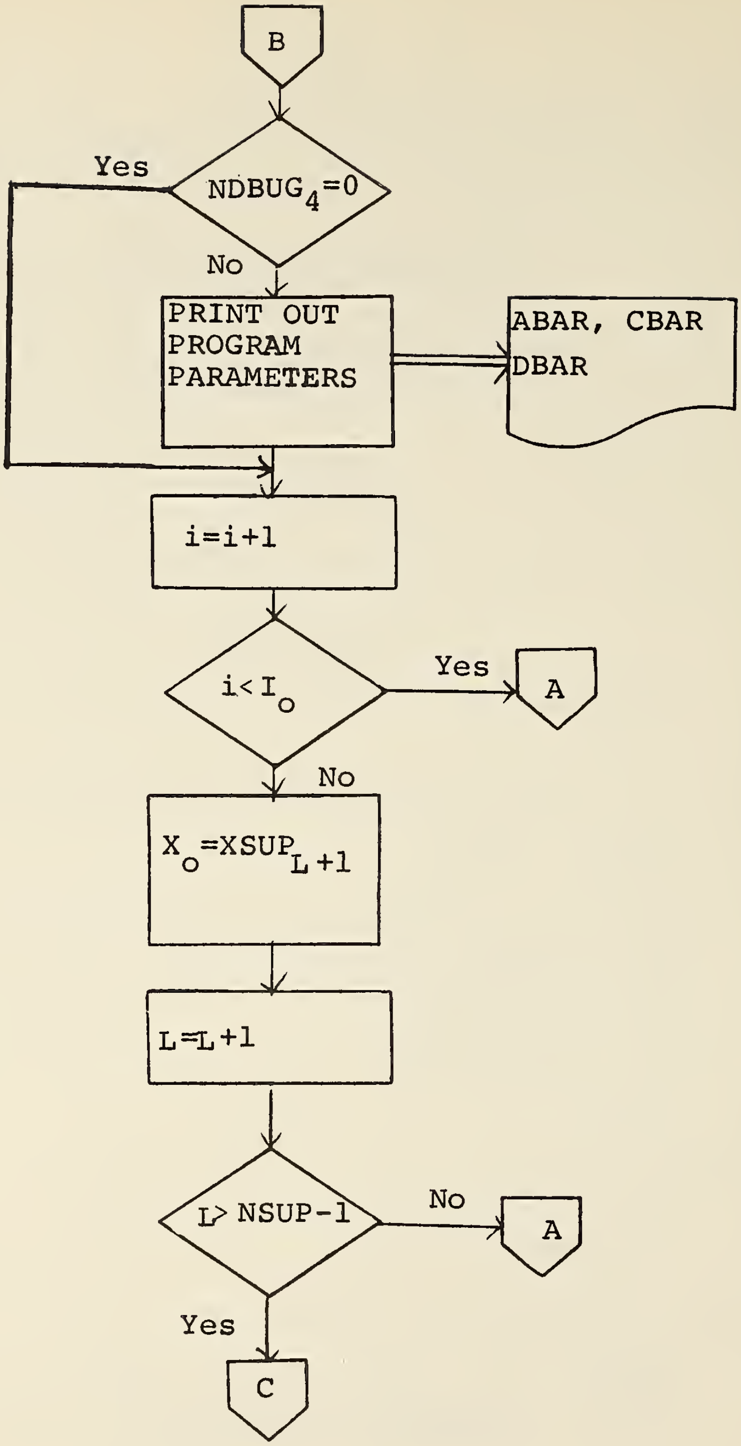


Figure 26. DMOMNT Program Flow Chart (Continued)

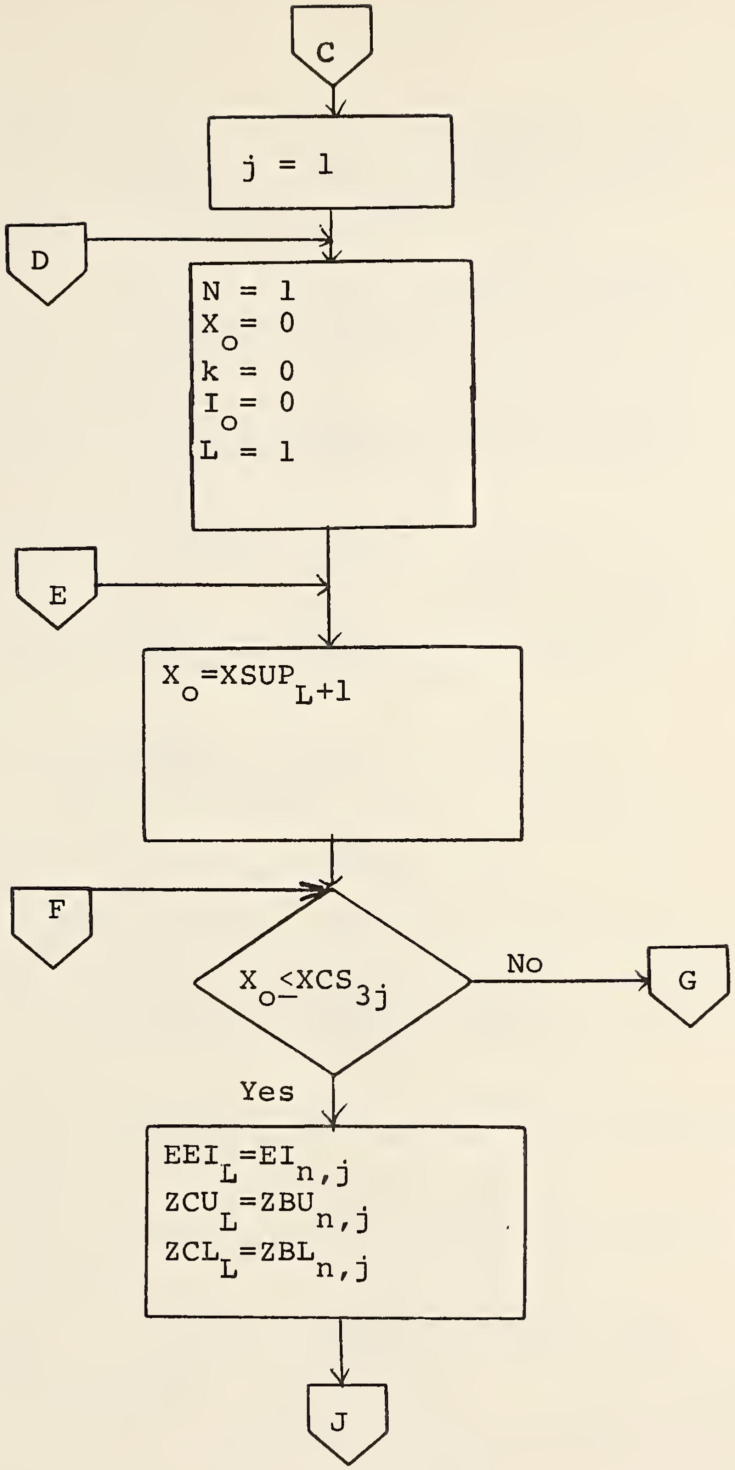


Figure 26. DMOMNT Program Flow Chart (Continued)

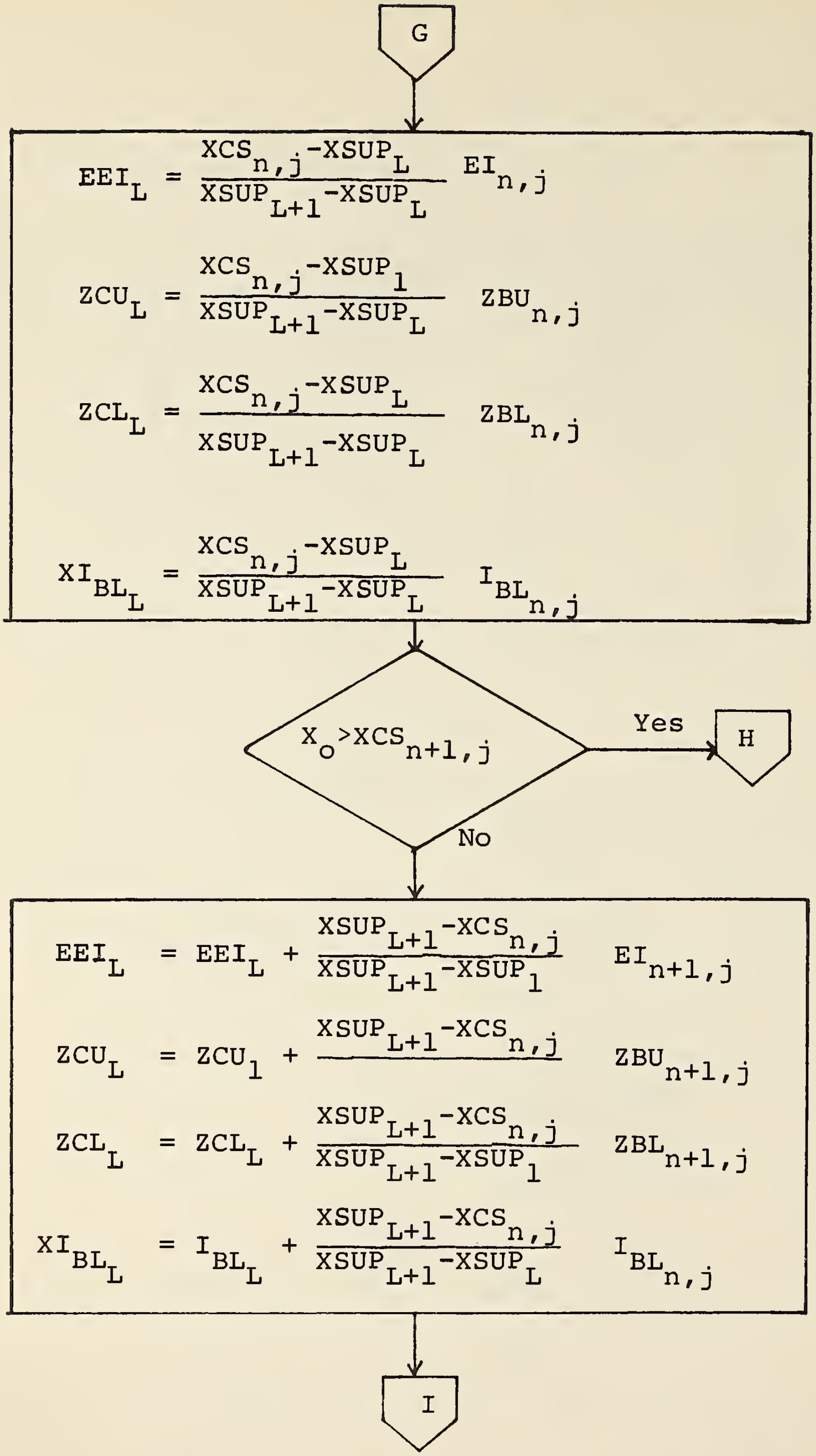

G
$EEI_L = \frac{XCS_{n,j} - XSUP_L}{XSUP_{L+1} - XSUP_L} EI_{n,j}$
$ZCU_L = \frac{XCS_{n,j} - XSUP_1}{XSUP_{L+1} - XSUP_L} ZBU_{n,j}$
$ZCL_L = \frac{XCS_{n,j} - XSUP_L}{XSUP_{L+1} - XSUP_L} ZBL_{n,j}$
$XI_{BL_L} = \frac{XCS_{n,j} - XSUP_L}{XSUP_{L+1} - XSUP_L} I_{BL_{n,j}}$
$X_o > XCS_{n+1,j}$
Yes
H
No
$EEI_L = EEI_L + \frac{XSUP_{L+1} - XCS_{n,j}}{XSUP_{L+1} - XSUP_1} EI_{n+1,j}$
$ZCU_L = ZCU_1 + \frac{XSUP_{L+1} - XCS_{n,j}}{} ZBU_{n+1,j}$
$ZCL_L = ZCL_L + \frac{XSUP_{L+1} - XCS_{n,j}}{XSUP_{L+1} - XSUP_1} ZBL_{n+1,j}$
$XI_{BL_L} = I_{BL_L} + \frac{XSUP_{L+1} - XCS_{n,j}}{XSUP_{L+1} - XSUP_L} I_{BL_{n,j}}$
I

Figure 26. DMOMNT Program Flow Chart (Continued)

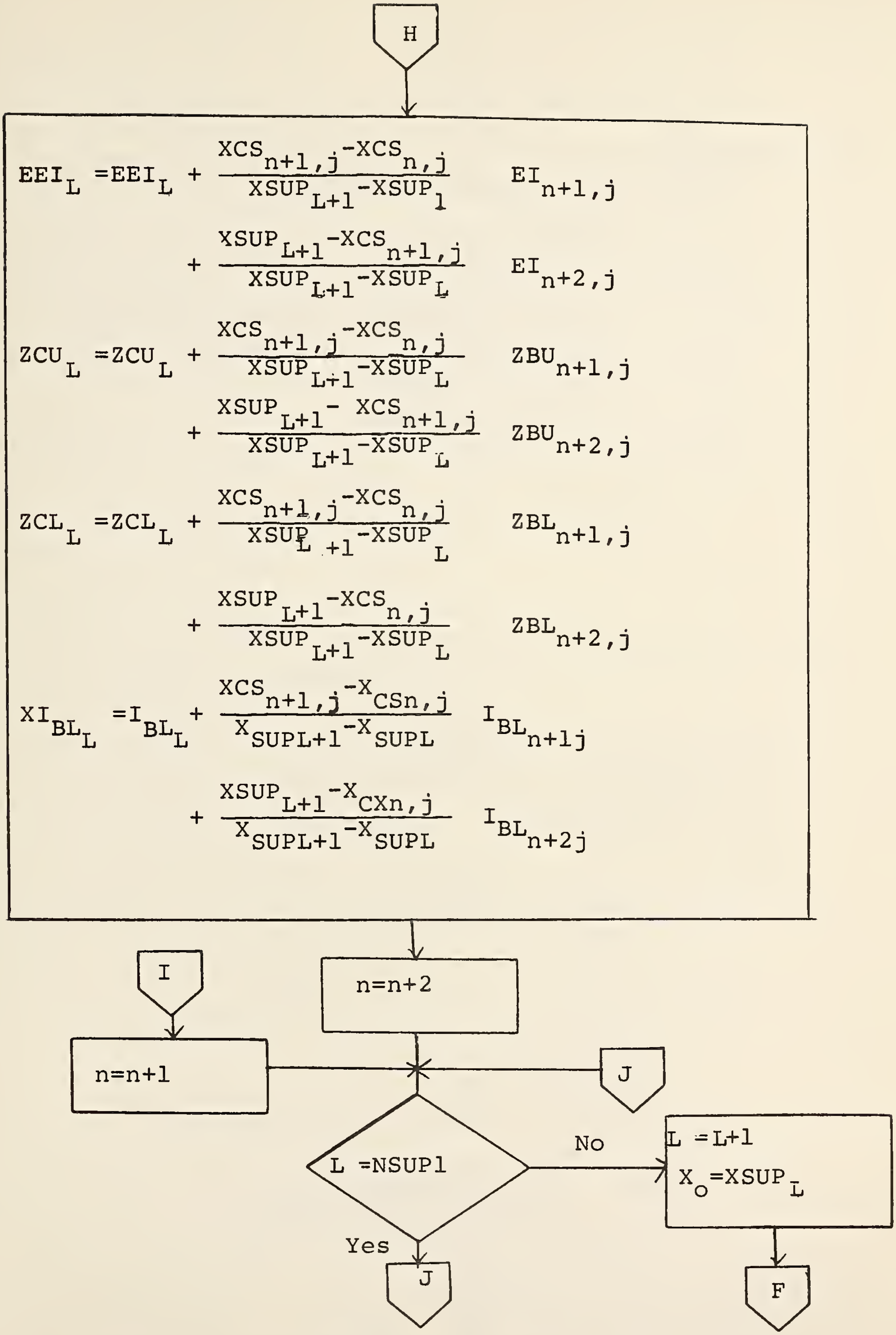

H
$EEI_L = EEI_L + \frac{XCS_{n+1,j} - XCS_{n,j}}{XSUP_{L+1} - XSUP_l} EI_{n+1,j} + \frac{XSUP_{L+1} - XCS_{n+1,j}}{XSUP_{L+1} - XSUP_L} EI_{n+2,j}$
$ZCU_L = ZCU_L + \frac{XCS_{n+1,j} - XCS_{n,j}}{XSUP_{L+1} - XSUP_L} ZBU_{n+1,j} + \frac{XSUP_{L+1} - XCS_{n+1,j}}{XSUP_{L+1} - XSUP_L} ZBU_{n+2,j}$
$ZCL_L = ZCL_L + \frac{XCS_{n+1,j} - XCS_{n,j}}{XSUP_{L+1} - XSUP_L} ZBL_{n+1,j} + \frac{XSUP_{L+1} - XCS_{n,j}}{XSUP_{L+1} - XSUP_L} ZBL_{n+2,j}$
$XI_{BL_L} = I_{BL_L} + \frac{XCS_{n+1,j} - X_{CSn,j}}{X_{SUPL+1} - X_{SUPL}} I_{BL_{n+1j}} + \frac{XSUP_{L+1} - X_{CXn,j}}{X_{SUPL+1} - X_{SUPL}} I_{BL_{n+2j}}$
I
n=n+1
n=n+2
J
L =NSUP1
No
$L = L+1$
$X_o = XSUP_L$
Yes
J
F

Figure 26. DMOMNT Program Flow Chart (Continued)

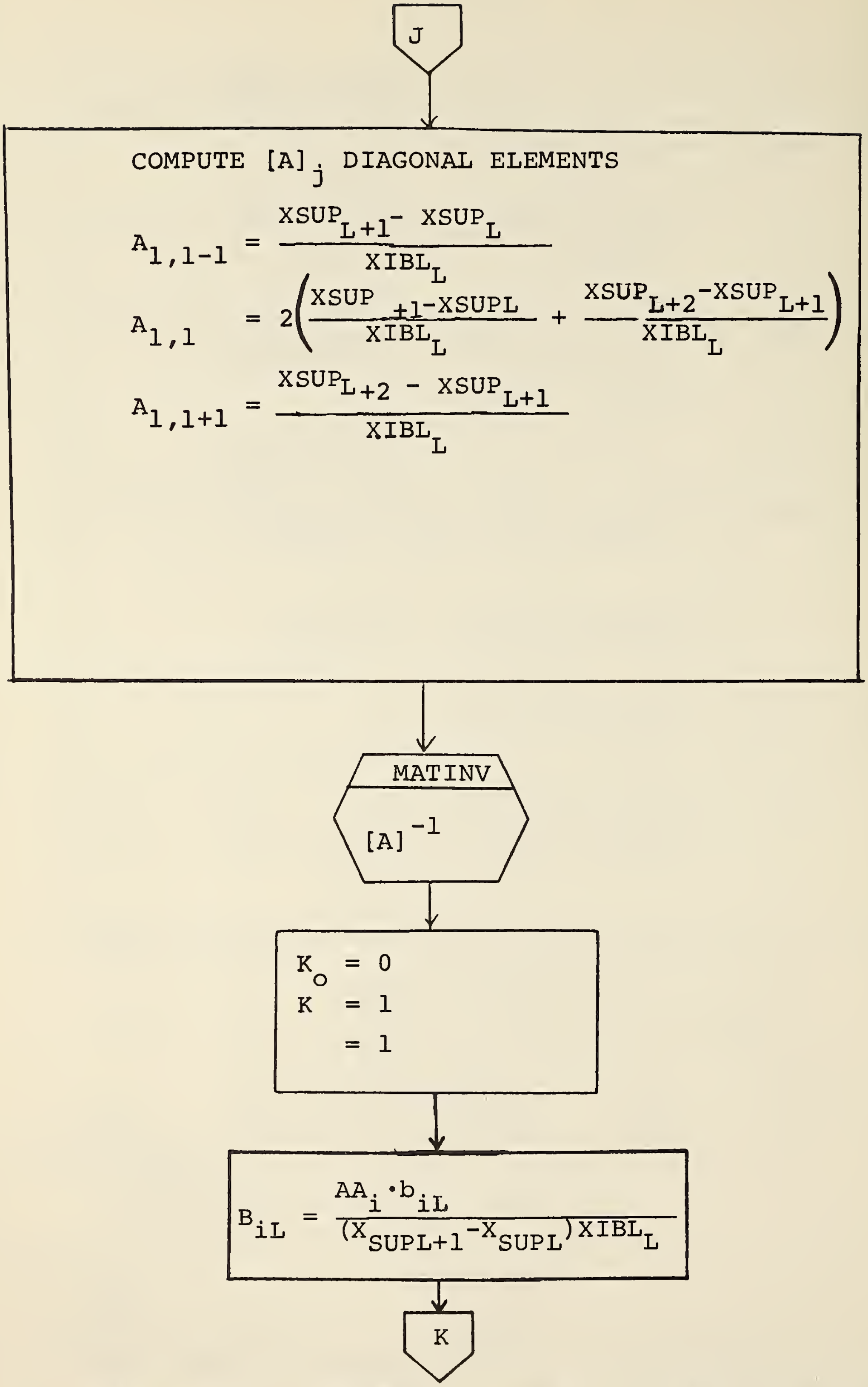

J
COMPUTE $[A]_j$ DIAGONAL ELEMENTS
$A_{1,1-1} = \frac{XSUP_{L+1} - XSUP_L}{XIBL_L}$
$A_{1,1} = 2\left(\frac{XSUP_{+1}-XSUPL}{XIBL_L} + \frac{XSUP_{L+2}-XSUP_{L+1}}{XIBL_L}\right)$
$A_{1,1+1} = \frac{XSUP_{L+2} - XSUP_{L+1}}{XIBL_L}$
MATINV
$[A]^{-1}$
$K_o = 0$
$K = 1$
$= 1$
$B_{iL} = \frac{AA_i \cdot b_{iL}}{(X_{SUPL+1} - X_{SUPL})XIBL_L}$
K

Figure 26. DMOMNT Program Flow Chart (Continued)

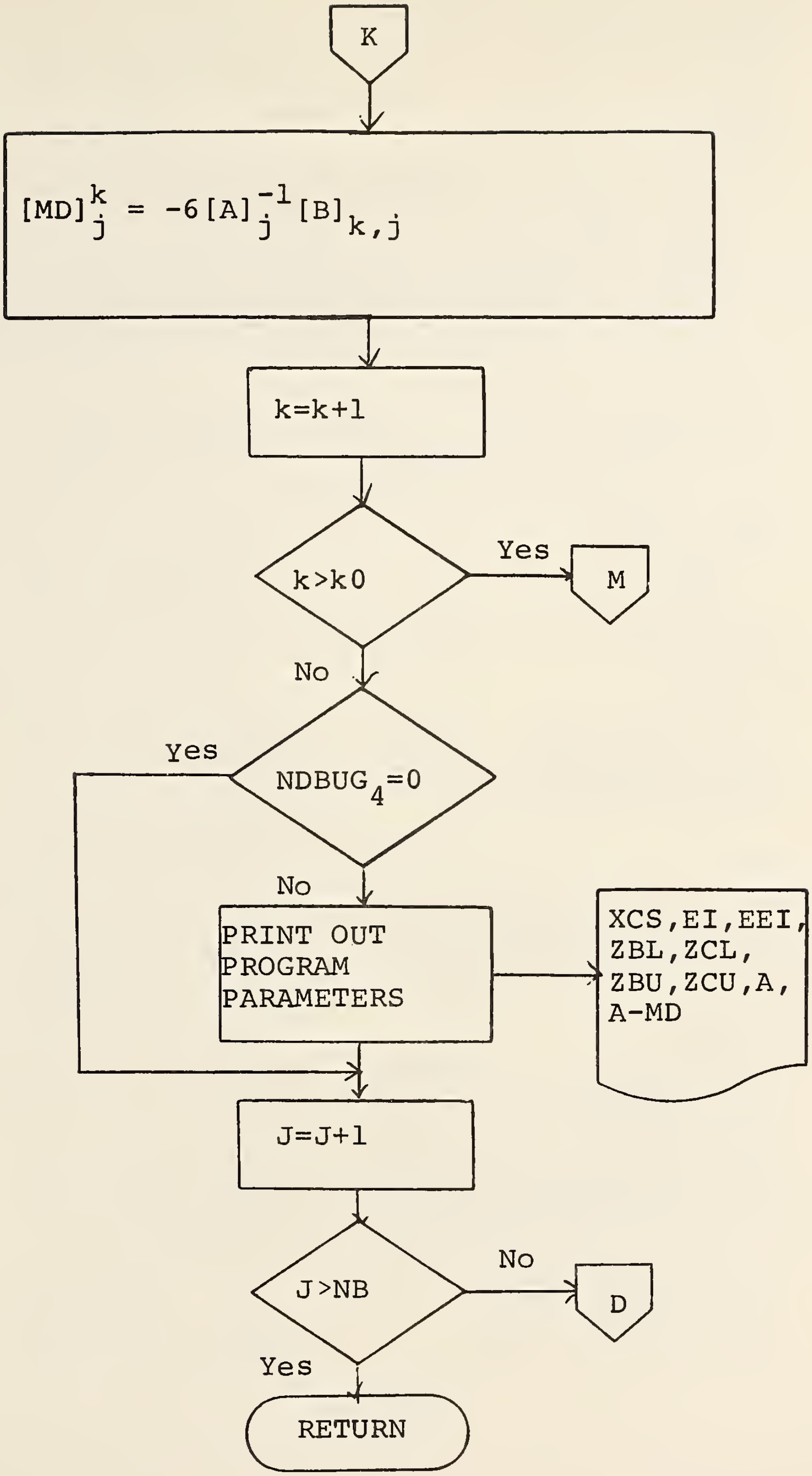

K
[MD]$_j^k$ = -6[A]$_j^{-1}$[B]$_{k,j}$
k=k+1
k>k0
Yes
M
No
Yes
NDBUG$_4$=0
No
PRINT OUT PROGRAM PARAMETERS
XCS,EI,EEI, ZBL,ZCL, ZBU,ZCU,A, A-MD
J=J+1
J>NB
No
D
Yes
RETURN

Figure 26. DMOMNT Program Flow Chart (Continued)

```
0001                SUBROUTINE DMOMNT
            C
0002                INTEGER P
0003                REAL IBL
0004                REAL MD, MDT
            C
0005                COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     (
                  1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                  2   NXPTS, NDBUG(10),
                  3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                  4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                  6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                  7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
            C
0006                COMMON /BLK6/ IBL(3,10)
            C
0007                DIMENSION XIBL(10,10), AA(20,10), SA(20,10),SB(20,10),
                  1      A(9,9), B(9), AINV(9,9)
            C
0008                DO 55 L=1,10
0009                DO 5  I=1,20
0010                  AA(I,L)  =  0.0
0011                  SA(I,L)  =  0.0
0012              5   SB(I,L)  =  0.0
0013                DO 54  I=1,10
0014             54 XIBL(I,L)  =  0.0
0015             55 CONTINUE
            C
0016                DO 6  J=1,9
0017                DO 6  I=1,9
0018                A(I,J)  = 0.0
0019              6 AINV(I,J)  =  0.0
            C
0020                IO  =  0
0021                I  =  1
            C
0022                IF (NDBUG(4).EQ.0) GO TO 15
0023                WRITE (IO,1005)
0024           1005 FORMAT ('ODMOMNT',///,'     L    I          X          XO        ',
                  1   '    A1            A2             AA             SA              SB')
0025             15 CONTINUE
0026                I1  =  0
0027                DO 20 L=1,NSI
0028                XO  =  XSUP(L)
0029                X1  =  XSUP(L+1)
0030                IO  =  I1  +  1
0031                    I1  =  IO  +  IS(L)-1
0032                DO 20  I=IO,I1
            C
0033             16 CONTINUE
0034                H = (X(I)-XO)/(X1-XO)*(X1-X(I))
0035                A2  =  H*(X1-X(I))*.5
0036                A1  =  (H/2.)*(X(I)-XO)
0037                AA(I,L)  =  A1     +  A2
```

```
0038                 XA1 = (X(I) + X0)/2.- X0
0039                 XB1 = (X(I) + X1)/2. - X0
0040                 HA = H/2.
0041                 HB = HA
0042                 XL = X1-X0
0043                 SA(I,L) = (XL*XB1)/(XL-XA1+XB1)
0044                 SB(I,L) = XL - SA(I,L)
           C
0045                 IF (NDBUG(4).EQ.0) GO TO 18
0046                 WRITE (IO,1018) L, I, X(I), X0, A1, A2, AA(I,L), SA(I,L), SB(I,L)
0047            1018 FORMAT (1X, 2I4, 2F10.3, 5E13.4)
0048              18 CONTINUE
           C
0049              20 CONTINUE
           C
0050                 DO 500 J=1,NB
0051                 N = 1
0052                 K = 0
0053                 IO = 0
           C
0054                 DO 150 L=1,NSI
0055                 X0 = XSUP(L+1)
0056                 X1 = XSUP(L)
           C
0057             100 IF(X0.LE.XCS(N,J))  GO TO 120
           C
0058                 XS = XSUP(L)
0059                 EEI(L,J) = (XCS(N,J) - XS)/(X0 - XS)*EI(N,J)
0060                 ZCL(L,J) = (XCS(N,J) - XS)/(X0 - XS)*ZBL(N,J)
0061                 ZCU(L,J) = (XCS(N,J) - XS)/(X0 - XS)*ZBU(N,J)
0062                 XIBL(L,J) = (XCS(N,J) - XS)/(X0-XS)*IBL(N,J)
0063                 XS =          XSUP(L+1)
0064                 IF(X0.GT.XCS(N+1,J)) GO TO 110
           C
0065                 EEI(L,J) = EEI(L,J) + (XS - XCS(N,J))/(X0 - X1)*EI(N+1,J)
0066                 ZCL(L,J) = ZCL(L,J) + (XS - XCS(N,J))/(X0 - X1)*ZBL(N+1,J)
0067                 ZCU(L,J) = ZCU(L,J) + (XS - XCS(N,J))/(X0 - X1)*ZBU(N+1,J)
0068                 XIBL(L,J) = XIBL(L,J) + (XS-XCS(N,J))/(X0-X1)*IBL(N+1,J)
0069                 N = N + 1
0070                 GO TO 150
           C
0071             110 EEI(L,J) = EEI(L,J) + (XCS(N+1,J) - XCS(N,J))/(X0 - X1) *EI(N+1,J)
                   1    + (XS - XCS(N+1,J))/(X0 - X1)*EI(N+2,J)
0072                 ZCL(L,J) = ZCL(L,J) + (XCS(N+1,J) - XCS(N,J))/(X0 - X1)*ZBL(N+1,J)
                   1    + (XS - XCS(N+1,J))/(X0 - X1)*ZBL(N+2,J)
0073                 ZCU(L,J) = ZCU(L,J) + (XCS(N+1,J) - XCS(N,J))/(X0 - X1)*ZBU(N+1,J)
                   1    + (XS - XCS(N+1,J))/(X0 - X1)*ZBU(N+2,J)
0074                 XIBL(L,J) = XIBL(L,J) + (XCS(N+1,J)-XCS(N,J))/(X0-X1)*IBL(N+1,J)
                   1      + (XS-XCS(N+1,J))/(X0-X1)*IBL(N+2,J)
0075                 N = N + 2
0076                 GO TO 150
0077             120 EEI(L,J) = EI(N,J)
0078                 ZCU(L,J) = ZBU(N,J)
0079                 ZCL(L,J) = ZBL(N,J)
```

```
      XIBL(L,J) = IBL(N,J)
C
  150 CONTINUE
C
      NSIM1 = NSI - 1
C
      DO 200 L=1,NSIM1
      DO 200 M=1,NSIM1
  200 A(L,M) = 0.0
C
      DO 210 L=1,NSIM1
      X0 = XSUP(L)
      X1 = XSUP(L+1)
      X2 = XSUP(L+2)
      IF(L.EQ.1)  GO TO 205
      A(L,L-1) = (X1 - X0)/XIBL(L,J)
C
  205 A(L,L) = 2*((X1 - X0)/XIBL(L,J) + (X2 - X1)/XIBL(L+1,J))
      IF(L.EQ.NSIM1)  GO TO 210
      A(L,L+1) = (X2 - X1)/XIBL(L+1,J)
  210 CONTINUE
C
      CALL MATINV (9,NSIM1, A, AINV, ADUM, AD, 3, IERR)
C
      I1=0
      DO 301 LN=1,NSIM1
      I0=I1+1
      I1=I0 + IS(LN) - 1
      DO 302 K=I0,I1
      DO 300 L=1,NSIM1
      IF (XSUP(L+1).LE.X(K))  GO TO 300
      IF (XSUP(L).GT.X(K))  GO TO 300
      B(L)=AA(K,L)*SB(K,L)/((XSUP(L+1)-XSUP(L))*XIBL(L,J))
  300 CONTINUE
C
      DO 351 LL = 1,NSIM1
  351 MD(LL+1,K,J)=0.
C
      DO 350 I=1,NSIM1
  350 MD(LL+1,K,J) = MD(LL+1,K,J) - 6. * AINV (LL,I)*B(I)
      MD(1,K,J)=0.
      MD(NSUP,K,J) = 0.
  302 CONTINUE
  301 CONTINUE
C
      IF (NDBUG(4).EQ.0) GO TO 160
      WRITE (IO,1150) J
 1150 FORMAT ('0BEAM =',I4)
      WRITE (IO,1151)
 1151 FORMAT ('0 XCS')
      WRITE (IO,1000) XCS
      WRITE (IO,1152)
 1152 FORMAT ('0 EI ')
      WRITE (IO,1000) EI
      WRITE (IO,1153)
 1153 FORMAT ('0 EEI')
      WRITE (IO,1000) EEI
      WRITE (IO,1154)
 1154 FORMAT ('0 ZBL')
      WRITE (IO,1000) ZBL
      WRITE (IO,1155)
 1155 FORMAT ('0 ZCL')
      WRITE (IO,1000) ZCL
      WRITE (IO,1156)
 1156 FORMAT ('0 ZBU')
      WRITE (IO,1000) ZBU
      WRITE (IO,1157)
 1157 FORMAT ('0 ZCU')
      WRITE (IO,1000) ZCU
      WRITE (IO,2020)
 2020 FORMAT('OXIBL')
      WRITE (IO,1001)   XIBL
      WRITE (IO,1158)
 1158 FORMAT ('0 A  ')
      WRITE (IO,1001) A
      WRITE (IO,1159)
 1159 FORMAT ('0 AINV')
      WRITE (IO,1001) AINV
 1001 FORMAT (1X,9E14.4)
 1000 FORMAT (1X,10E13.3)
      WRITE (IO,1060)
      WRITE (IO,1001)  B
 1060 FORMAT('0B')
C
      WRITE (IO,1160)
 1160 FORMAT ('0 MD')
      DO 170 K=1,20
      WRITE (IO,1170) (MD(LL,K,J),LL=1,11)
 1170 FORMAT (1X,11E11.3)
  170 CONTINUE
C
  160 CONTINUE
C
  500 CONTINUE
C
      RETURN
      END
```

PNTLOD

The subroutine PNTLOD calculates the vehicle tracks across the bridge:

$$\left.\begin{aligned} Y_1' &= Y_{CL1} - \frac{\Delta Y_T}{2} \\ Y_2' &= Y_{CL1} + \frac{\Delta Y_T}{2} \end{aligned}\right\}\quad \text{Lane 1}$$

$$\left.\begin{aligned} Y_3' &= Y_{CL2} - \frac{\Delta Y_T}{2} \\ Y_4' &= Y_{CL2} + \frac{\Delta Y_T}{2} \end{aligned}\right\}\quad \text{Lane 2}$$

Axle loads are distributed laterally to the tracks to which they apply.

$$\left.\begin{aligned} W_{i,1} &= \frac{W_{ii}}{2} \\ W_{i,2} &= \frac{W_{ii}}{2} \end{aligned}\right\}\quad \text{Axle in Lane 1}$$

$$\left.\begin{aligned} W_{i,3} &= \frac{W_{ii}}{2} \\ W_{i,4} &= \frac{W_{ii}}{2} \end{aligned}\right\}\quad \text{Axle in Lane 2}$$

where i is the load index (only axles actually on the bridge) and ii is the input axle load index and where $i \leq ii$.

No subroutines are called by this routine.

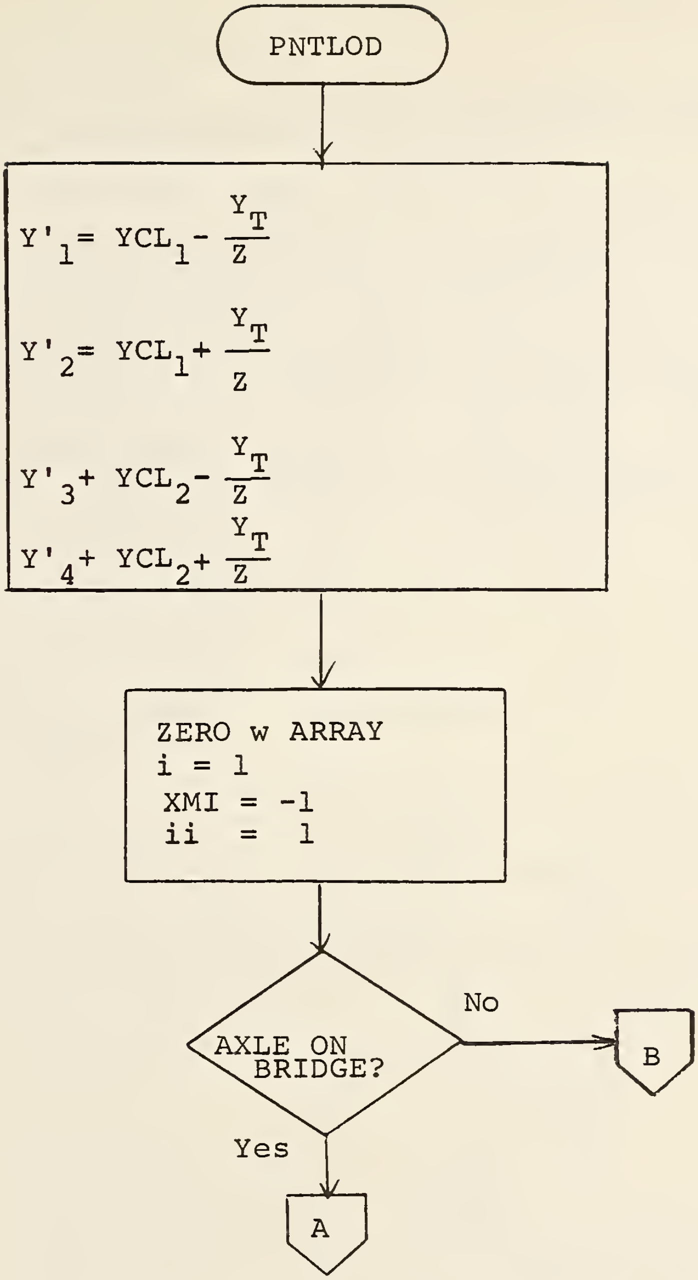

PNTLOD
$Y'_1 = YCL_1 - \frac{Y_T}{Z}$
$Y'_2 = YCL_1 + \frac{Y_T}{Z}$
$Y'_3 + YCL_2 - \frac{Y_T}{Z}$
$Y'_4 + YCL_2 + \frac{Y_T}{Z}$
ZERO w ARRAY
i = 1
XMI = -1
ii = 1
AXLE ON BRIDGE?
No
B
Yes
A

Figure 27. PNTLOD Program Flow Chart

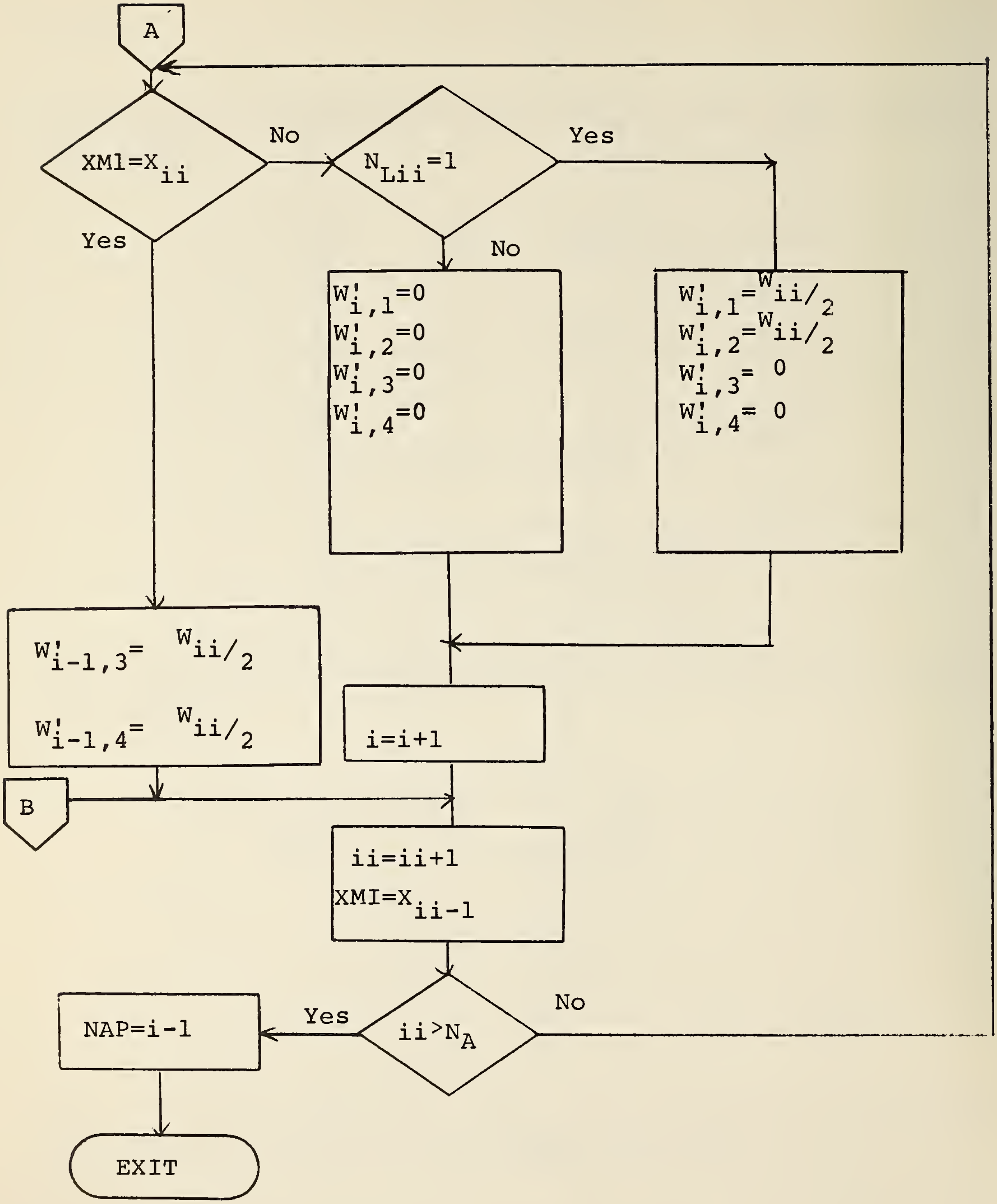
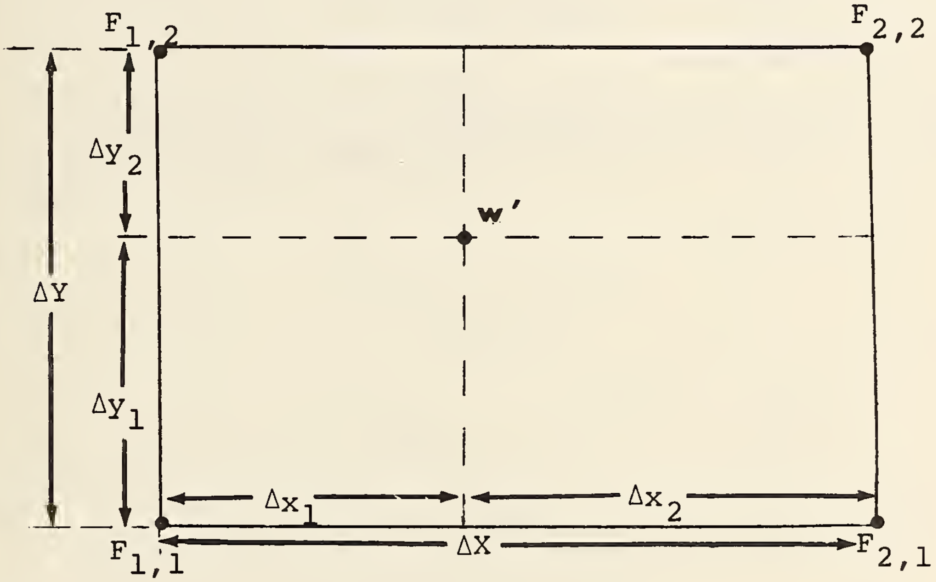


Figure 27. PNTLOD Program Flow Chart (Continued)

```
      SUBROUTINE PNTLOD
C
C     THIS SUBROUTINE ESTABLISHES THE LOCATION AND WEIGHT OF THE POINT
C      LOADS REPRESENTED BY THE INCOMING DYNAMIC LOAD DATA
C
      REAL MD, MDT
C
      COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     (
     1  NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
     2  NXPTS, NDBUG(10),
     3  DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
     4  XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
     6  E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
     7  F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
C
      COMMON /BLK3/ SUMHRX, DTL, IEVNTX, NA,     NTRUKX, LAST,
     1     LTYPE(20), WGTX(20),SPDX(20), LLANE(20), TIMEX(20),
     2     XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
      DYT=3.0
C
      YP(1)=YCL(1)-DYT
      YP(2)=YCL(1)+DYT
      YP(3)=YCL(2)-DYT
      YP(4)=YCL(2)+DYT
C
      XM1 = -1.0
      DO 100 J=1,4
      DO 100 I=1,50
  100 WP(I,J) = 0.0
C
      I=1
      DO 1 II=1,NA
      IF (XPOSX(II).LT.0.0.OR.XPOSX(II).GT.XLB)  GO TO 1
C
      IF (XM1.EQ.XPOSX(II)) GO TO 2
      IF (LNUMX(II).EQ.2) GO TO 3
      WP(I,1)=WEITX(II)/2.
      WP(I,2)=WEITX(II)/2.
      WP(I,3)=0.0
      WP(I,4)=0.0
      GO TO 4
C
    3 WP(I,1)=0.0
      WP(I,2)=0.0
      WP(I,3)=WEITX(II)/2.
      WP(I,4)=WEITX(II)/2.
    4 CONTINUE
      XP(I) = XPOSX(II)
      I=I+1
      GO TO 5
C
    2 WP(I-1,3)=WEITX(II)/2.
      WP(I-1,4)=WEITX(II)/2.
    5 XM1=XPOSX(II)
    1 CONTINUE
      NAP=I-1
C
      IF (NDBUG(5).EQ.0)  GO TO 30
      WRITE (IO,10) NA,NAP
   10 FORMAT ('0 PNTLOD  NA=',I4,'  NAP=',I4)
      WRITE (IO,1010)
 1010 FORMAT ('0YP  ')
      WRITE (IO,11) YP
   11 FORMAT (1X,4E16.4)
      WRITE (IO,1011)
 1011 FORMAT ('0 XP ')
      WRITE (IO,12) XP
   12 FORMAT (1X,10E11.3)
      WRITE (IO,1012)
 1012 FORMAT ('0 XPOSX')
      WRITE (IO,12) XPOSX
C
      WRITE (IO,1013)
 1013 FORMAT ('0 WP ')
      DO 6 I=1,NAP
    6 WRITE (IO,13) (WP(I,J),J=1,4),I
   13 FORMAT (1X,4E16.4,I6)
   30 CONTINUE
C
C
      RETURN
      END
```

GRDLOD

The GRDLOD routine calculates the force vector F due to the point loads **w′** on the bridge deck. The point loads are distributed to the nearest grid points on the basis of area. The components of the force vector due to one load are calculated as



$$F_{1,1} = -w_i' \left(1 - \frac{\Delta x_1 \cdot \Delta y_1}{\Delta X \cdot \Delta Y} \right)$$

$$F_{1,2} = -w_i' \left(1 - \frac{\Delta x_1 \cdot \Delta y_2}{\Delta X \cdot \Delta Y} \right)$$

$$F_{2,1} = -w_i' \left(1 - \frac{\Delta x_2 \cdot \Delta y_1}{\Delta X \cdot \Delta Y} \right)$$

$$F_{2,2} = -w_i' \left(1 - \frac{\Delta x_2 \cdot \Delta y_2}{\Delta X \cdot \Delta Y} \right)$$

All forces acting on the same point are summed for that point.

If the span is a simple span the bending moments are calculated as

$$MB_{m,j} = \sum_{i=1}^{IS_1} \left(XSUP_2 - X_m\right) \left(\frac{F_{i,j} - (XSUP_2 - X_i)}{XSUP_2} \right)$$

at X_m for a load at X_i.
and the static bending stresses $SIGB_{i,j}$ are calculated by dividing $MB_{i,j}$ by the lower cross sectional modulus $ZBL_{n,j}$, (below the neutral axis).

No subroutines are called by this routine.

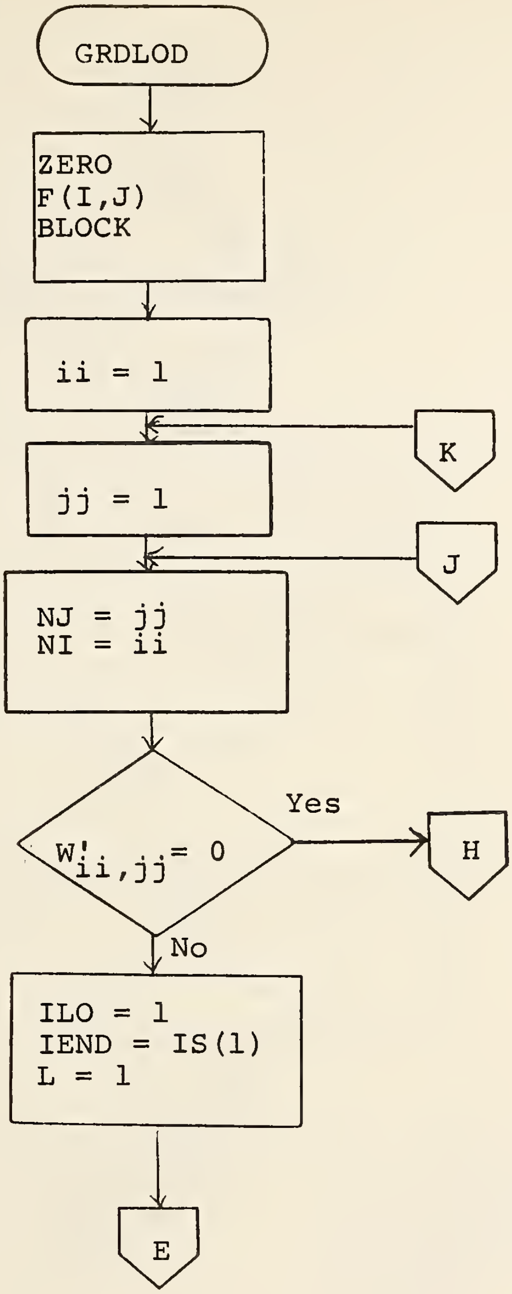


Figure 28. GRDLOD Program Flow Chart

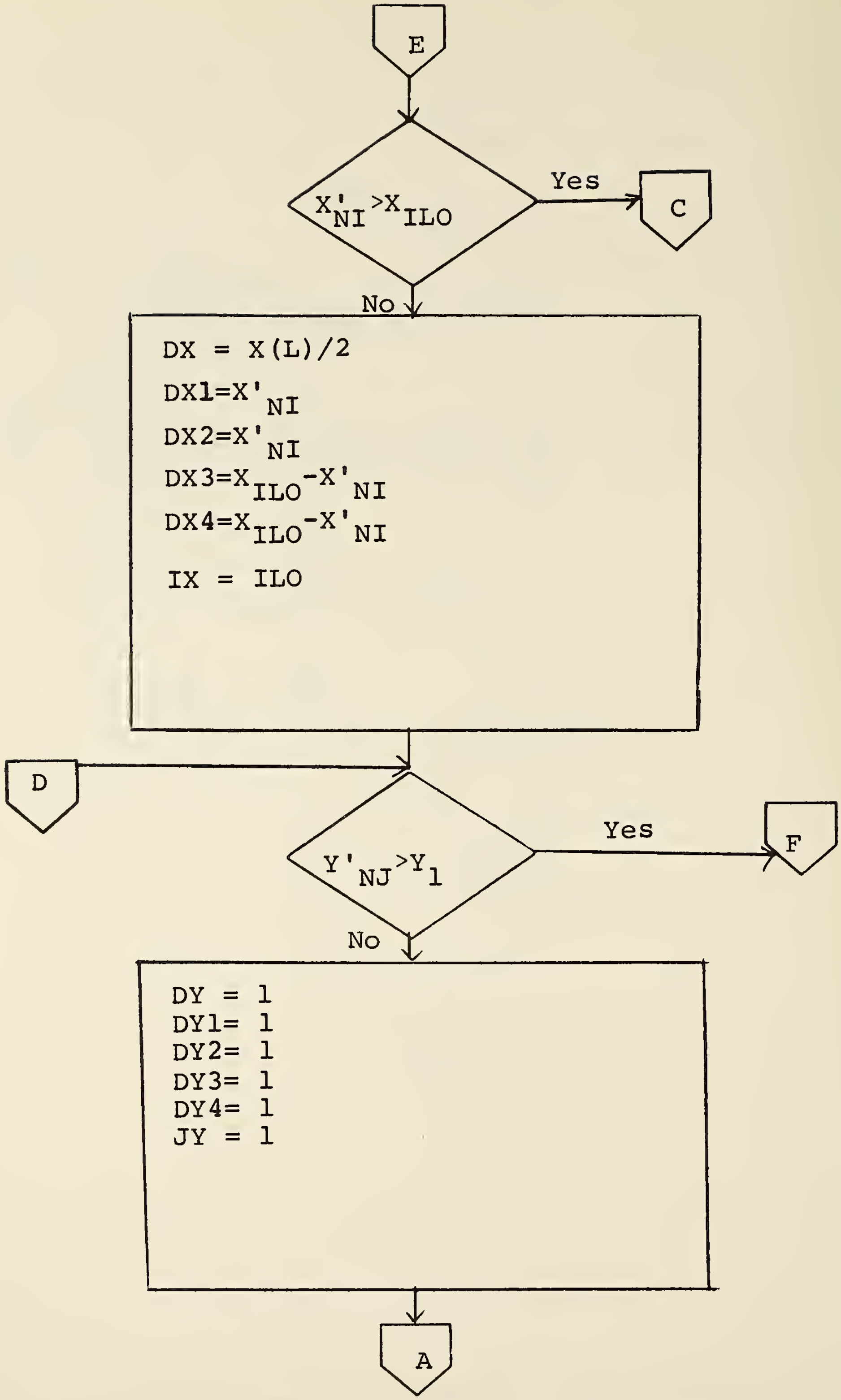


Figure 28. GRDLOD Program Flow Chart (Continued)

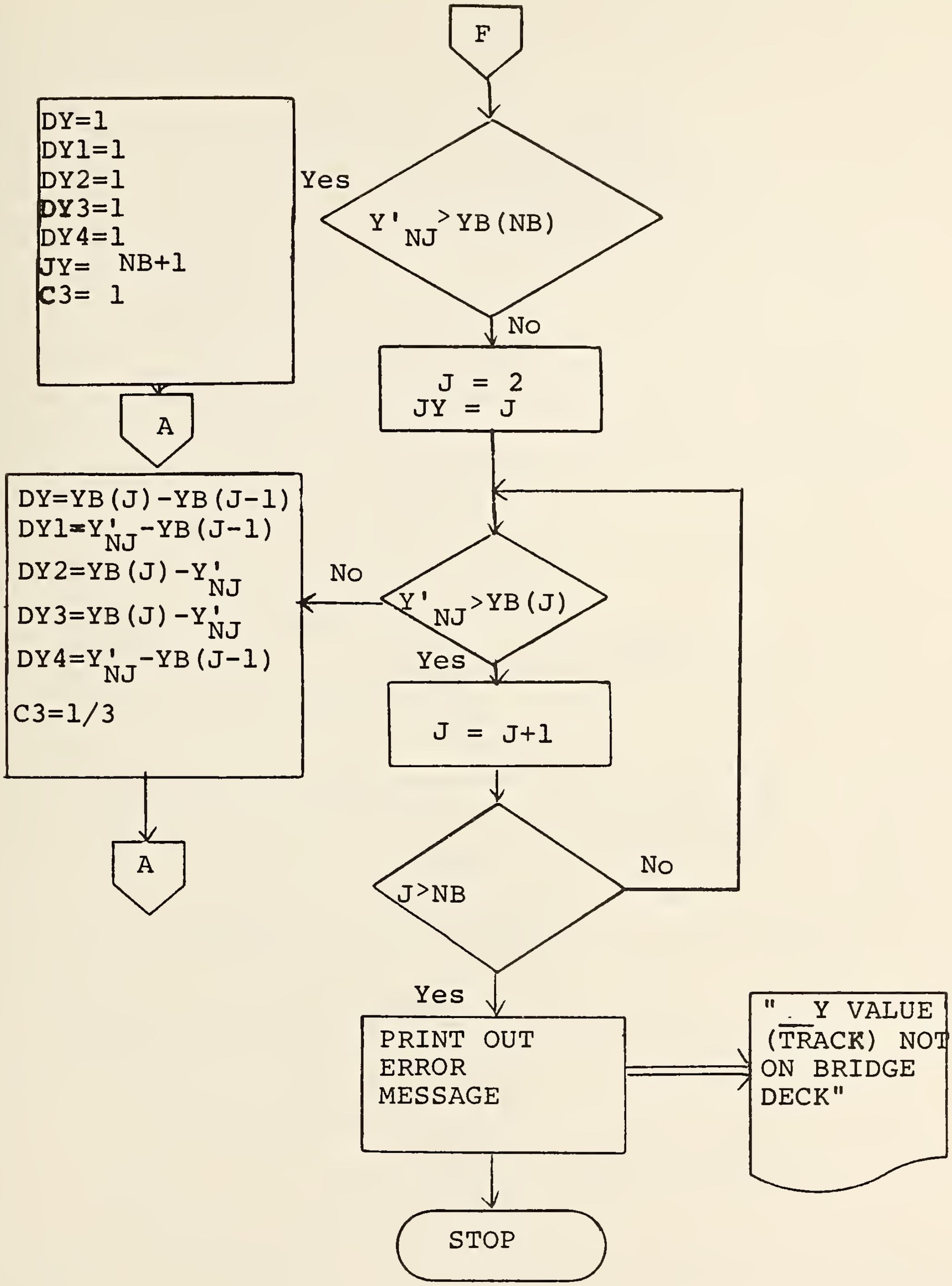


Figure 28. GRDLOD Program Flow Chart (Continued)

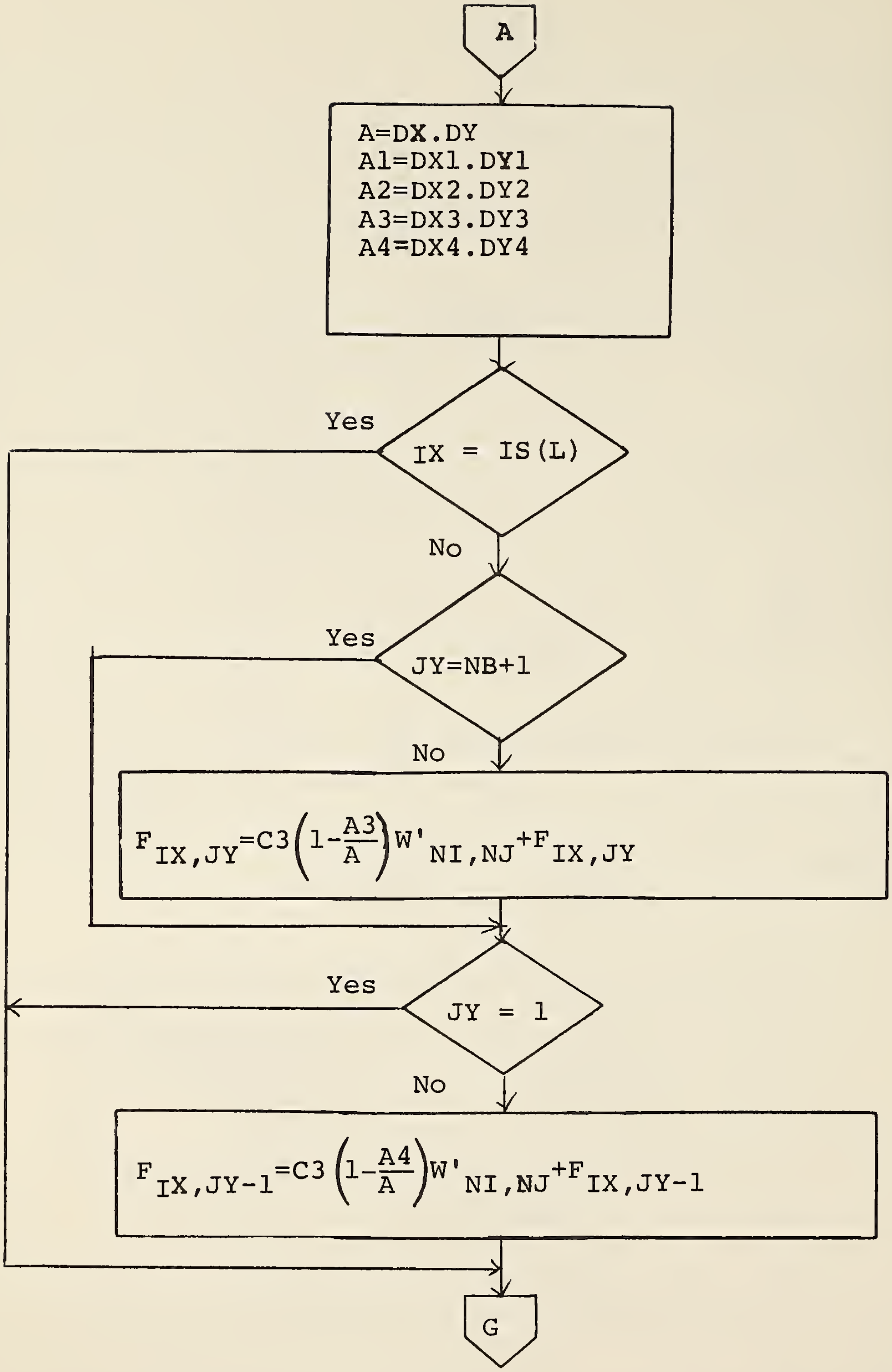
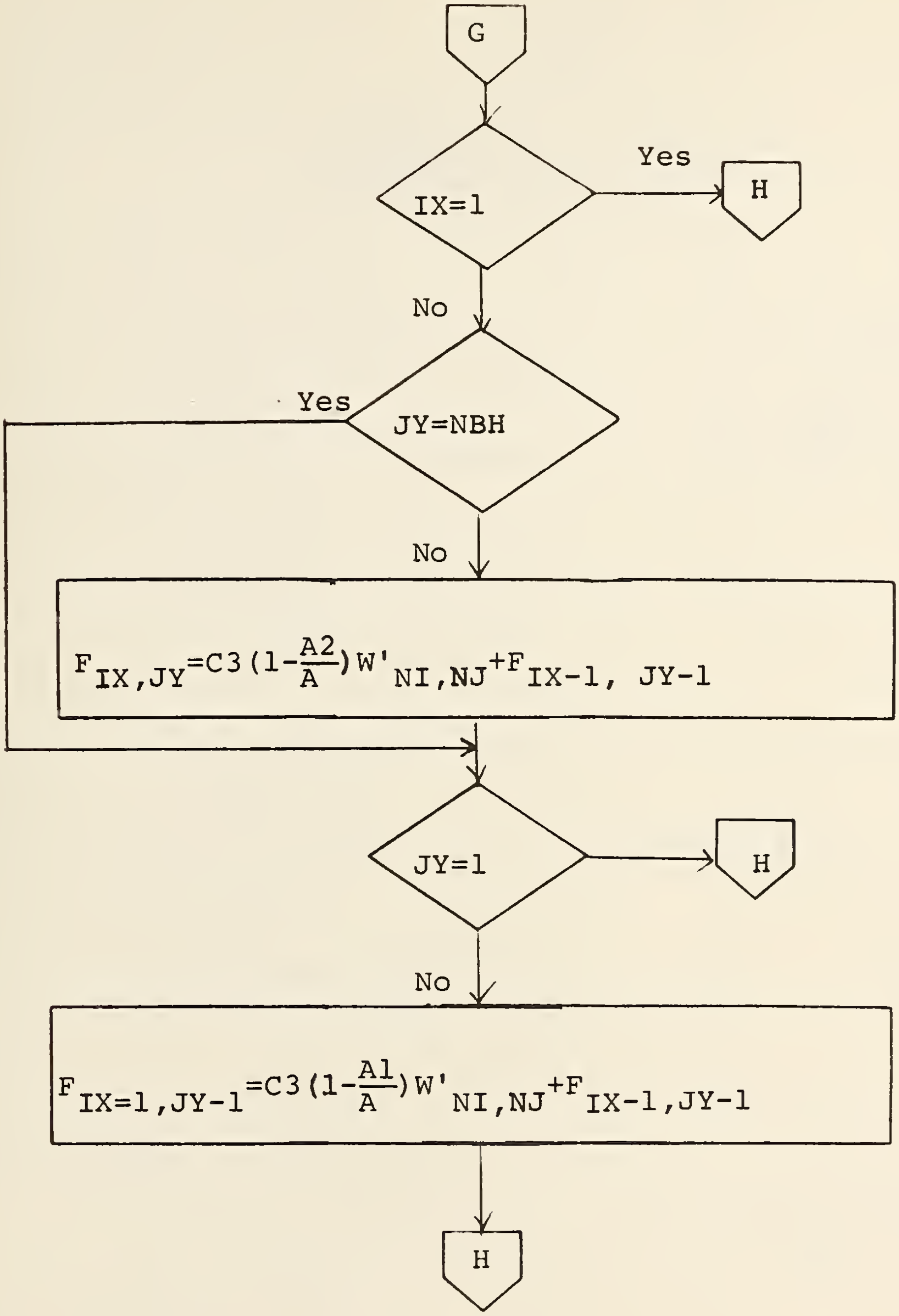

A
A=DX.DY
A1=DX1.DY1
A2=DX2.DY2
A3=DX3.DY3
A4=DX4.DY4
Yes
IX = IS(L)
No
Yes
JY=NB+1
No
$F_{IX,JY}=C3\left(1-\frac{A3}{A}\right)W'_{NI,NJ}+F_{IX,JY}$
Yes
JY = 1
No
$F_{IX,JY-1}=C3\left(1-\frac{A4}{A}\right)W'_{NI,NJ}+F_{IX,JY-1}$
G

Figure 28. GRDLOD Program Flow Chart (Continued)



G
IX=1
Yes
H
No
JY=NBH
Yes
No
$F_{IX,JY}=C3(1-\frac{A2}{A})W'_{NI,NJ}+F_{IX-1,\ JY-1}$
JY=1
H
No
$F_{IX=1,JY-1}=C3(1-\frac{A1}{A})W'_{NI,NJ}+F_{IX-1,JY-1}$
H

Figure 28. GRDLOD Program Flow Chart (Continued)

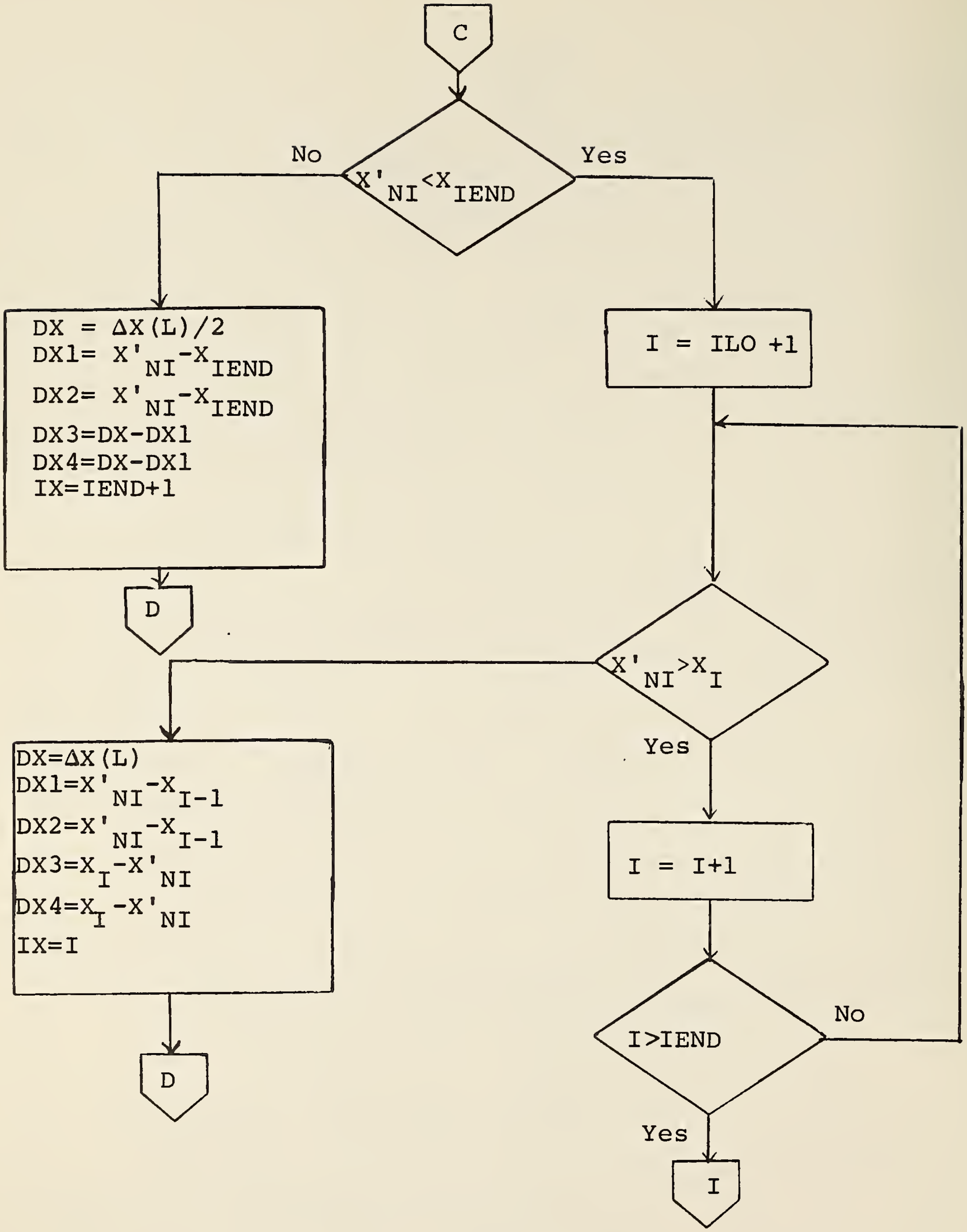


Figure 28. GRDLOD Program Flow Chart (Continued)

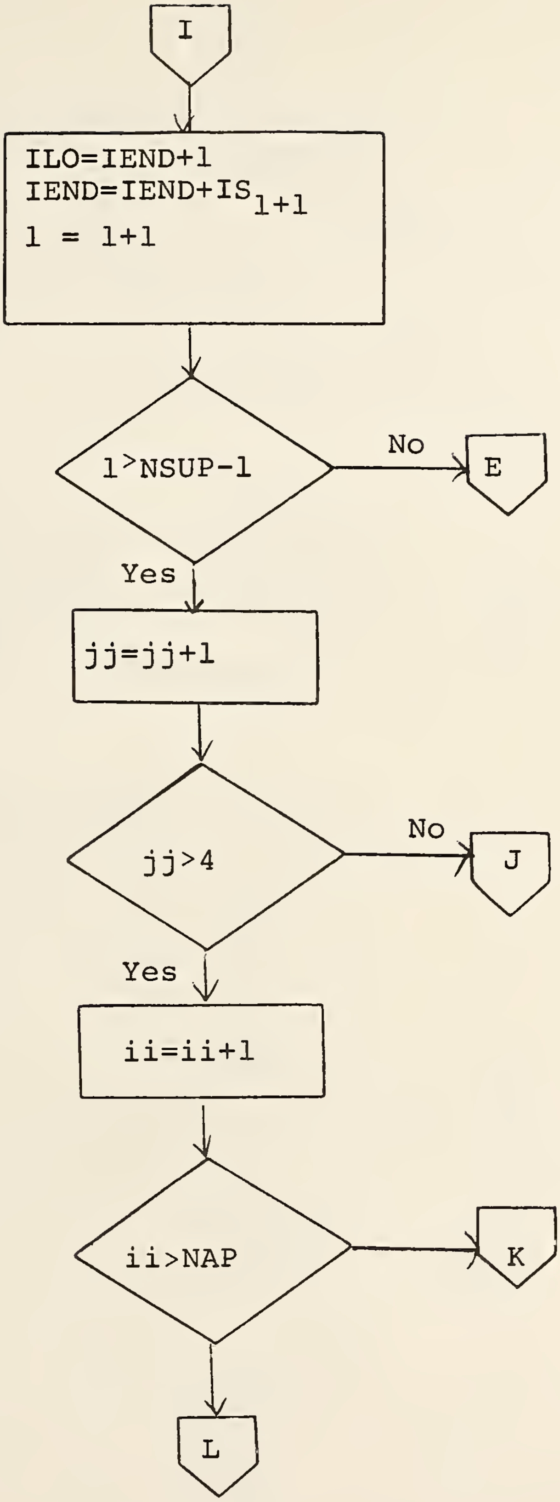


Figure 28. GRDLOD Program Flow Chart (Continued)

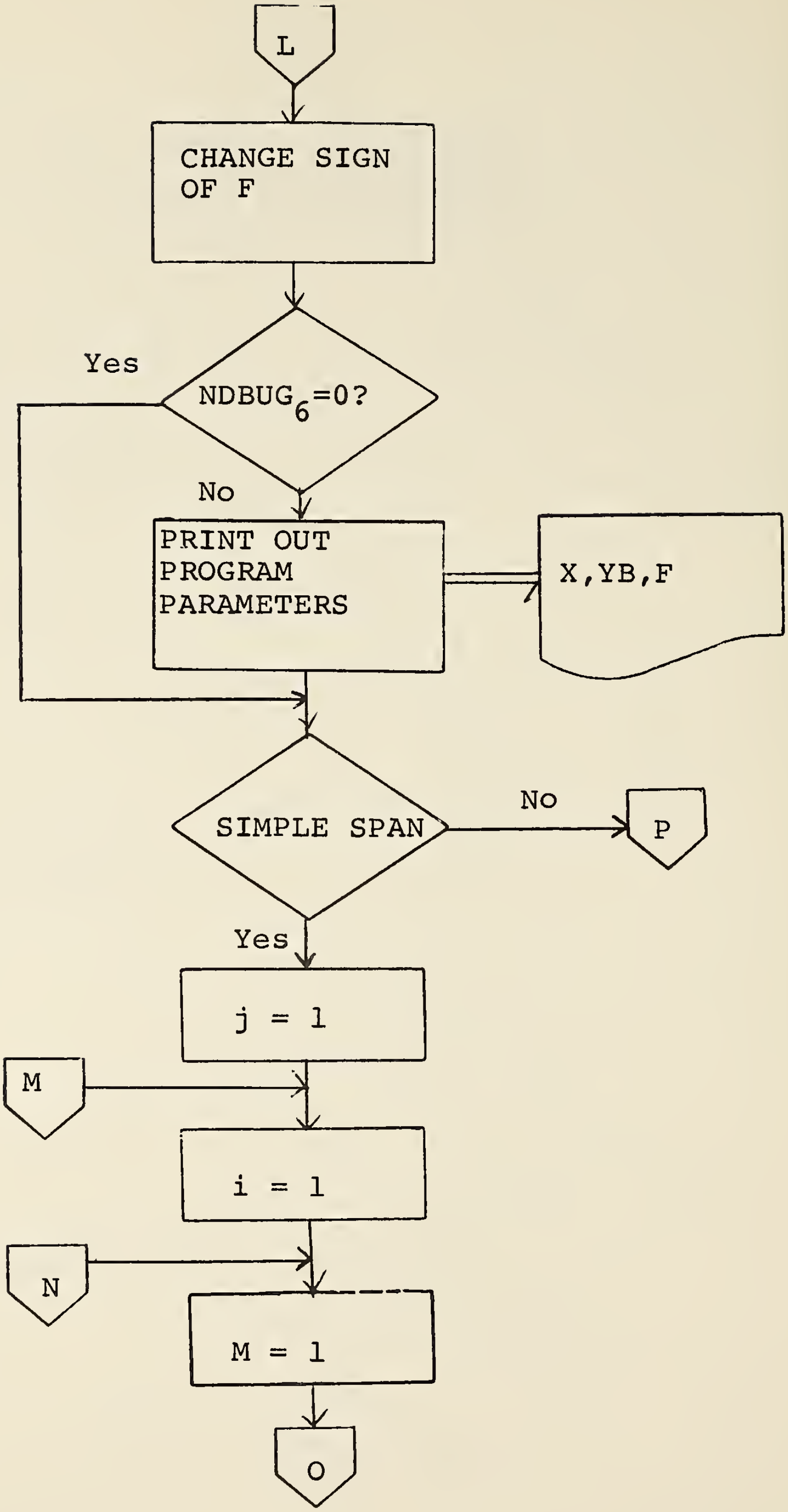
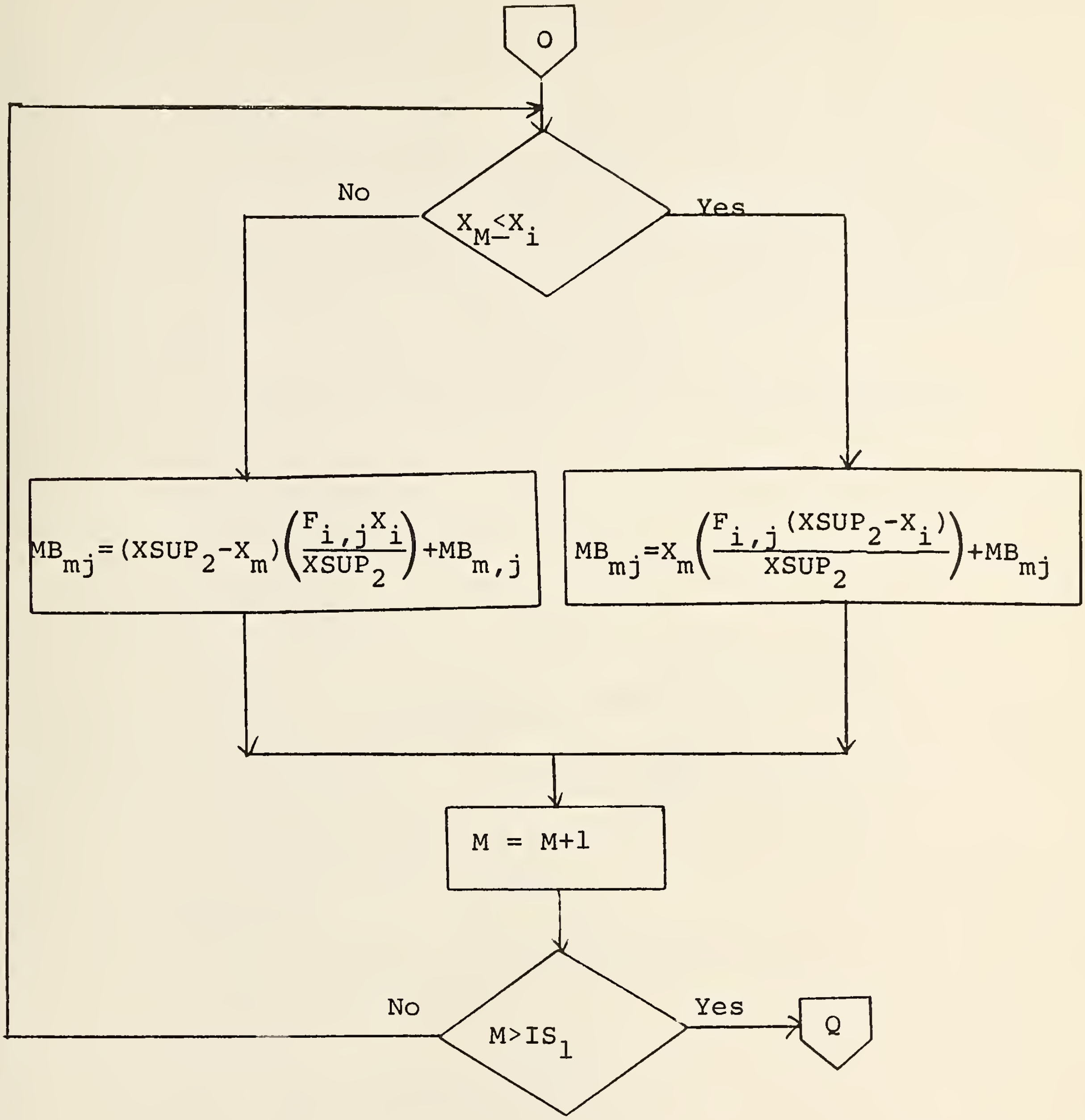


Figure 28. GRDLOD Program Flow Chart (Continued)



O
No
$X_M \le X_i$
Yes
$MB_{mj} = (XSUP_2 - X_m)\left(\frac{F_{i,j} X_i}{XSUP_2}\right) + MB_{m,j}$
$MB_{mj} = X_m\left(\frac{F_{i,j}(XSUP_2 - X_i)}{XSUP_2}\right) + MB_{mj}$
M = M+1
No
$M > IS_1$
Yes
Q

Figure 28. GRDLOD Program Flow Chart (Continued)

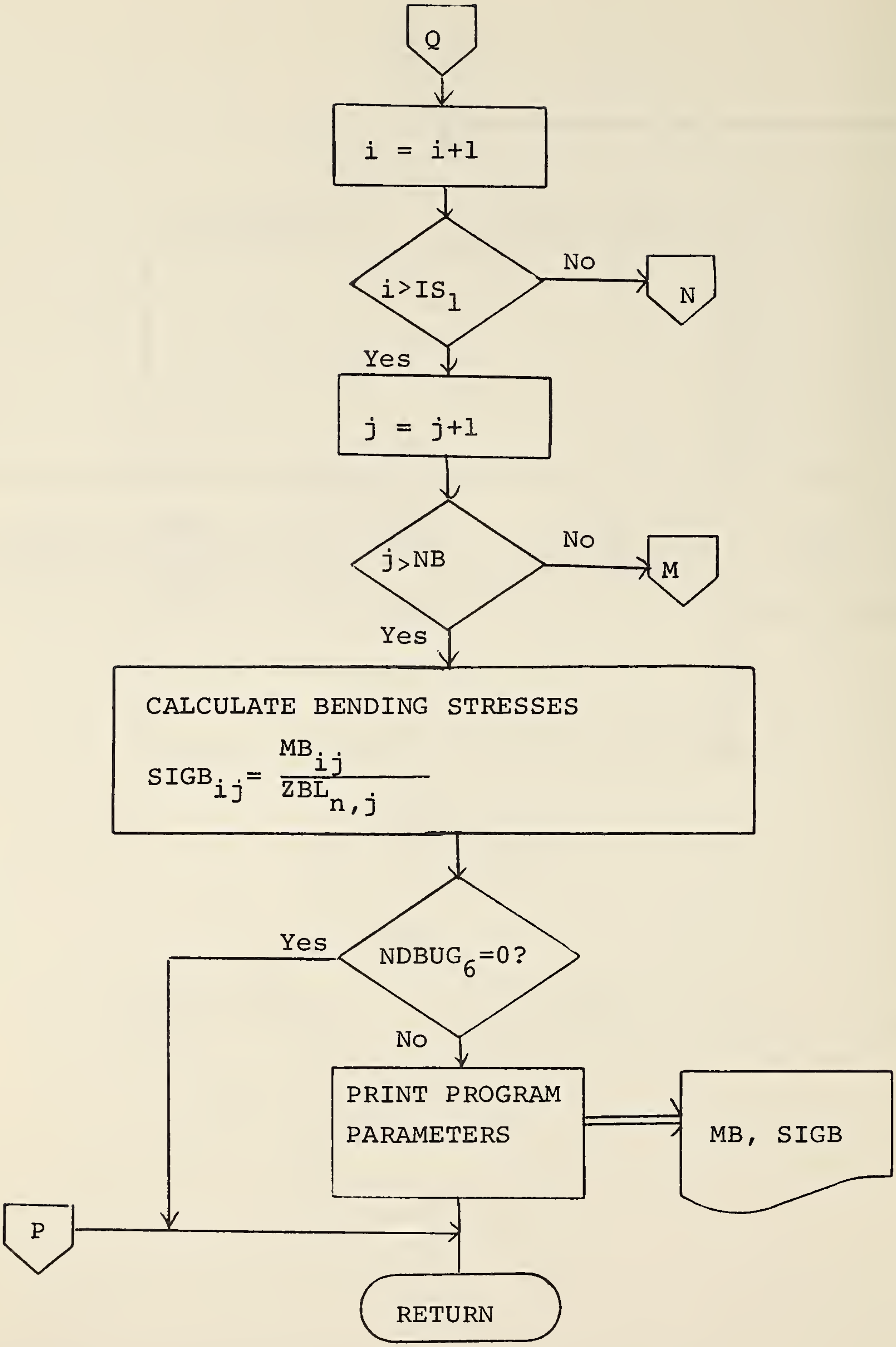


Figure 28. GRDLOD Program Flow Chart (Continued)

```
      SUBROUTINE GRDLOD
C
C     THIS SUBROUTINE CONVERTS POINT LOADS TO GRID LOADS
C
      REAL MD, MDT,MB
C
      COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,      0
     1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
     2   NXPTS, NDBUG(10),
     3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
     4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
     6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
     7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
C
      COMMON /BLK4/ XK(80,80),XKI(80,80), ATEMP(80,80), DTEMP(80)
     1          ,MI(20), MB(20), SIGB(20,10)
C
      DIMENSION SK(20,20,10), SKP(10,10)
      EQUIVALENCE (SK(1,1,1),ATEMP(1,1)), (SKP(1,1),ATEMP(1,51))
C
      ILO=1
      IEND=IS(1)
      DO 1 L=1,NSI
      DO 2 I=ILO,IEND
      DO 2 J=1,NB
    2 F(I,J)=0.0
      ILO=IEND+1
    1 IEND=IEND+IS(L+1)
      DO 3 II=1,NAP
      DO 4 JJ=1,4
      NJ=JJ
      NA=II
      IF (WP(II,JJ).EQ.0.0) GO TO 4
      ILO=1
      IEND=NXPTS
      IF (XP(NA).GT.X(ILO)) GO TO 10
      IL1 = 0
      DO 5 L=1,NSI
      IL1 = IL1 + IS(L)
      IF (ILO.LT.IL1)  GO TO 55
    5 CONTINUE
   55 CONTINUE
      DX = DELX(L)/2.
      DX1=XP(NA)
      DX2=DX1
      DX3=X(ILO) - XP(NA)
      DX4=DX3
      IX=ILO
C
   11 IF (YP(NJ).GT.YB(1)) GO TO 6
      DY=1.
      DY1=1.
      DY4=1.
      DY2=1.0
      DY3=1.0
      C3 = 1.0
      JY=1
    7 A=DX*DY
      A1=DX1*DY1
      A2=DX2*DY2
      A3=DX3*DY3
      A4=DX4*DY4
C
      IF (IX.EQ.NXPTS+1)  GO TO 80
      IF (JY.EQ.NB+1)  GO TO 81
      F(IX,JY)=(1.-A3/A)*WP(NA,NJ)*C3+F(IX,JY)
   81 IF (JY.EQ.1)  GO TO 80
      F(IX,JY-1)=(1.-A4/A)*WP(NA,NJ)*C3+F(IX,JY-1)
   80 IF (IX.EQ.1)  GO TO 4
      IF (JY.EQ.NB+1)  GO TO 82
      F(IX-1,JY)=(1.-A2/A)*WP(NA,NJ)*C3+F(IX-1,JY)
   82 IF (JY.EQ.1)  GO TO 4
      F(IX-1,JY-1)=(1.-A1/A)*WP(NA,NJ)*C3+F(IX-1,JY-1)
      GO TO 4
C
    6 IF (YP(NJ).LE.YB(NB)) GO TO 8
      DY=1.
      DY1=1.0
      DY4=1.0
      DY2=1.0
      DY3=1.0
      C3 = 1.0
      JY=NB+1
      GO TO 7
    8 JO=2
      DO 9 J=JO,NB
      JY=J
      IF (YP(NJ).GT.YB(J)) GO TO 9
      DY=YB(J)-YB(J-1)
      DY1=YP(NJ)-YB(J-1)
      DY2=YB(J)-YP(NJ)
      DY3=YB(J)-YP(NJ)
      DY4=YP(NJ)-YB(J-1)
      C3 = 1.0/3.0
      GO TO 7
    9 CONTINUE
      WRITE (IO,100)
  100 FORMAT (1X,' Y VALUE (TRACK) NOT ON BRIDGE DECK ')
      STOP
C
   10 IF (XP(NA).LT.X(IEND)) GO TO 12
      DX=DELX(NSI)/2.
      DX1=XP(NA) - X(IEND)
      DX2=DX1
      DX3=DX - DX1
      DX4=DX3
      IX=IEND+1
      GO TO 11
```

```
          12 ILO=ILO+1
             DO 13 I=ILO,IEND
             IF (XP(NA).GT.X(I)) GO TO 13
             IL1 = 0
             DO 75 L=1,NSI
             IL1 = IL1 + IS(L)
             IF (I   .LT.IL1)  GO TO 76
          75 CONTINUE
          76 CONTINUE
             DX=DELX(L)
             DX1=XP(NA)-X(I-1)
             DX2=XP(NA)-X(I-1)
             DX3=X(I)-XP(NA)
             DX4=X(I)-XP(NA)
             IX=I
             GO TO 11
          13 CONTINUE
           4 CONTINUE
           3 CONTINUE
             IF (NDBUG(6).EQ.0) GO TO19
             WRITE (IO,102)
         102 FORMAT (1X,'GRDLOD')
             NA=0
             DO 14 L=1,NSI
          14 NA=NA+IS(L)
             DO 15 I=1,NA
             DO 15 J=1,NB
          15 WRITE (IO,103) X(I),YB(J),F(I,J)
         103 FORMAT (1X,3E16.4)
          19 CONTINUE
      C
      C   SIMPLE SPAN BENDING MOMENTS AND BENDING STRESSES
      C
             IF (NSUP.GT.2) GO TO 50
             IS1 = IS(1)
      C
             DO 45 J=1,NB
             DO 20 I=1,IS1
             SIGB(I,J) = 0.0
             MB(I)    = 0.
          20 CONTINUE
             DO 30 I=1,IS1
             TANTHA = (F(I,J)*(XSUP(2) - X(I)))/XSUP(2)
             TANTHB = (F(I,J)*X(I))/XSUP(2)
             DO 25 M=1,IS1
             IF (X(M).LE.X(I))  GO TO 23
             MB(M)    = (XSUP(2)-X(M))*TANTHB + MB(M)
             GO TO 25
      C
          23 MB(M)    = X(M)*TANTHA + MB(M)
          25 CONTINUE
          30 CONTINUE
      C
             DO 45 I=1,IS1
      NU = NCS(J)
      DO 41  N=1,NU
      IF(X(I).LE.XCS(N,J))  GO TO 42
   41 CONTINUE
      STOP 14
C
   42 CONTINUE
C
      SIGB(I,J) = (MB(I)   /ZBL(N,J))/144.0
      IF (NDBUG(6).EQ.0)  GO TO 45
      WRITE (IO,2045) I,J,N,MB(I)   ,ZBL(N,J),SIGB(I,J)
 2045 FORMAT (' I J N ',3I6,'      MB  ZBL   SIGB',3E12.4)
      WRITE (IO,1040)
      WRITE (IO,1041)  MB
   45 CONTINUE
      IF (NDBUG(6).EQ.0)  GO TO 50
      WRITE (IO,1042)
      WRITE (IO,1043)   SIGB
 1040 FORMAT ('0 MB    ')
 1041 FORMAT (1X,10E13.5)
 1042 FORMAT ('0 SIGB ')
 1043 FORMAT (1X,10E13.5)
   50 CONTINUE
C
      RETURN
      END
```

LODDIS

This subroutine develops the actual discontinuity moments, i.e., due to the imposed live load. The constant, during a given run, unit load discontinuity moment vectors are operated on by the applied force vector, i.e.,

$$\left[M_{DT}\right]_j = \sum_{i=1}^{IS(L)} \left\{F_i\right\}_j \left[M_D\right]_{ij}$$

No subroutines are called by this routine.

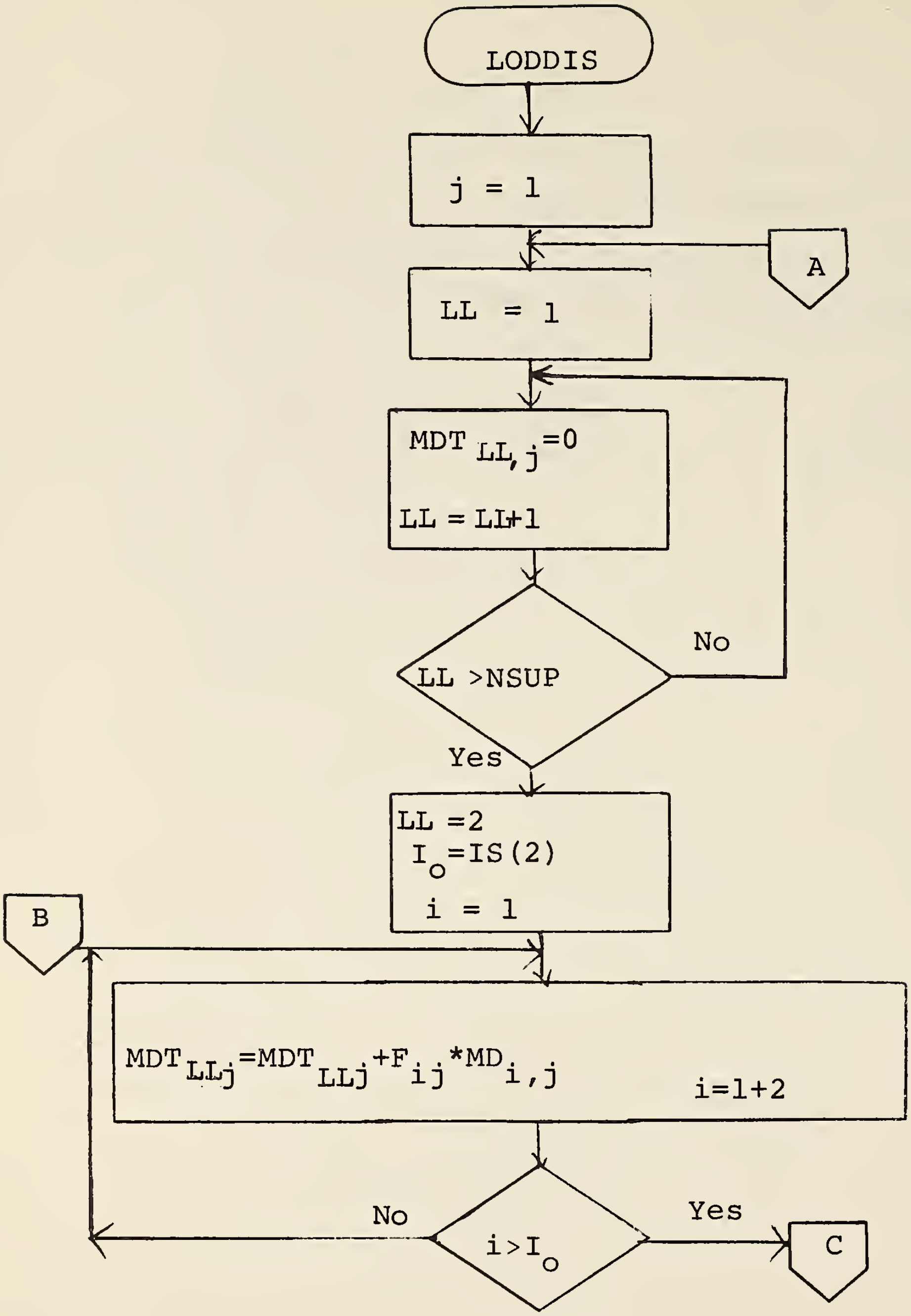


Figure 29. LODDIS Program Flow Chart

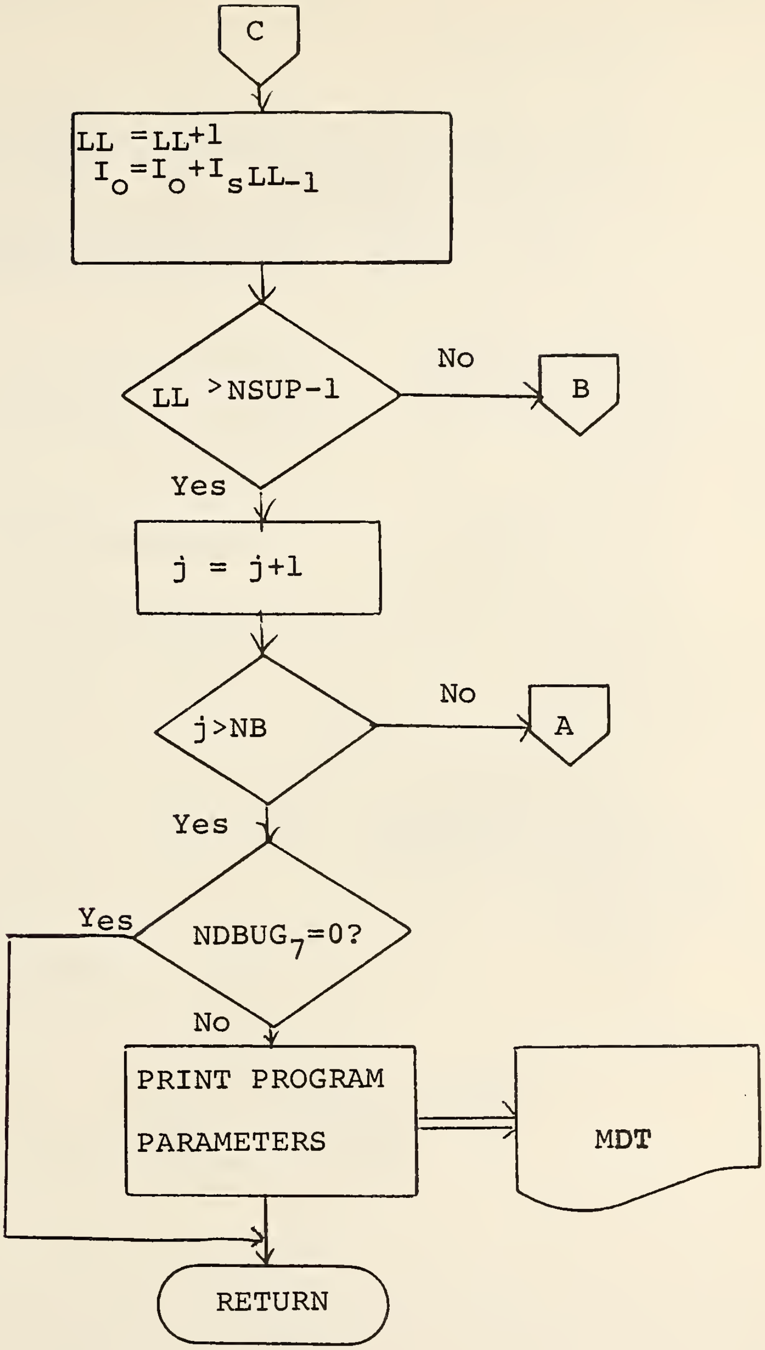


Figure 29. LODDIS Program Flow Chart

```
0001                SUBROUTINE LODDIS
0002                REAL MD, MDT
              C
0003                COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,
                  1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                  2   NXPTS, NDBUG(10),
                  3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                  4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                  6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                  7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
              C
0004                DO 50  J=1,NB
              C
0005                DO 10  LL=1,NSUP
0006             10 MDT(LL,J) = 0.0
              C
0007                IO = 1
0008                I1 = 0
              C
0009                DO 30 LL=2,NSI
0010                IO = IO + I1
0011                I1 = I1 + IS(LL-1)
0012                DO 20 I=IO,I1
0013                MDT(LL,J) = MDT(LL,J) + F(I,J) * MD(LL,I,J)
0014             20 CONTINUE
              C
0015             30 CONTINUE
0016             50 CONTINUE
              C
0017                IF (NDBUG(7).EQ.0)  GO TO 60
0018                WRITE (IO,1001)
0019           1001 FORMAT ('OMDT')
0020                WRITE (IO,1002) MDT
0021           1002 FORMAT (1X,11E11.3)
0022             60 CONTINUE
              C
0023                RETURN
0024                END
```

ASEMBL

The subroutine ASEMBL calculates:

The internal moments

$$MI_{i,j,L} = \sum\left(X_i\ F_{i,j}\left(\frac{X_i - X_0}{XSUP_{L+1}}\right)\right)$$

The bending moments

$$MB_{i,j,L} = MI_{i,j} + \left(1- \frac{X_i - X_0}{XSUP_{L+1}-XSUP_L}\right) MDT_{L,j} - \left(\frac{X_i - X_0}{XSUP_{L+1}-XSUP_L}\right) MDT_{L+1,j}$$

Deflections

$$\delta_{i,j} = \sum\left(\frac{X_i\ MB_{i,j}\ \Delta X_L}{EEI_{Lj}}\right)$$

The subroutine STIFF is called to calculate the longitudinal and lateral deflections, include the discontinuity contribution, to assemble the flexibility matrix per span (free body).

The following subroutines are called:

STIFF

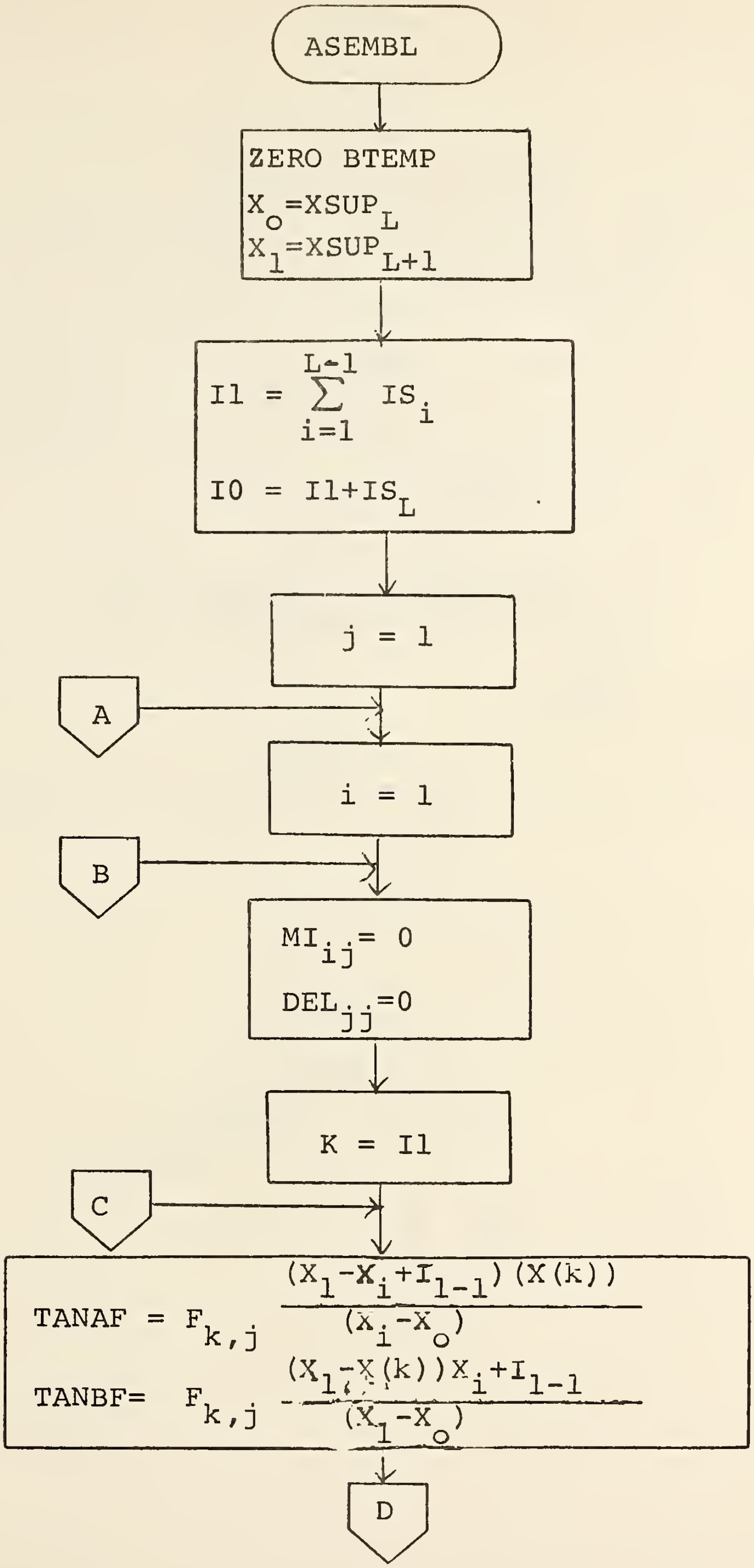

ASEMBL
ZERO BTEMP
X_o=XSUP$_L$
X_1=XSUP$_{L+1}$
$I1 = \sum_{i=1}^{L-1} IS_i$
$I0 = I1+IS_L$
$j = 1$
A
$i = 1$
B
$MI_{ij} = 0$
$DEL_{jj}=0$
$K = I1$
C
$TANAF = F_{k,j} \frac{(X_1-X_i+I_{1-1})(X(k))}{(X_i-X_o)}$
$TANBF = F_{k,j} \frac{(X_1-X(k))X_i+I_{1-1}}{(X_1-X_o)}$
D

Figure 30. ASEMBL Program Flow Chart

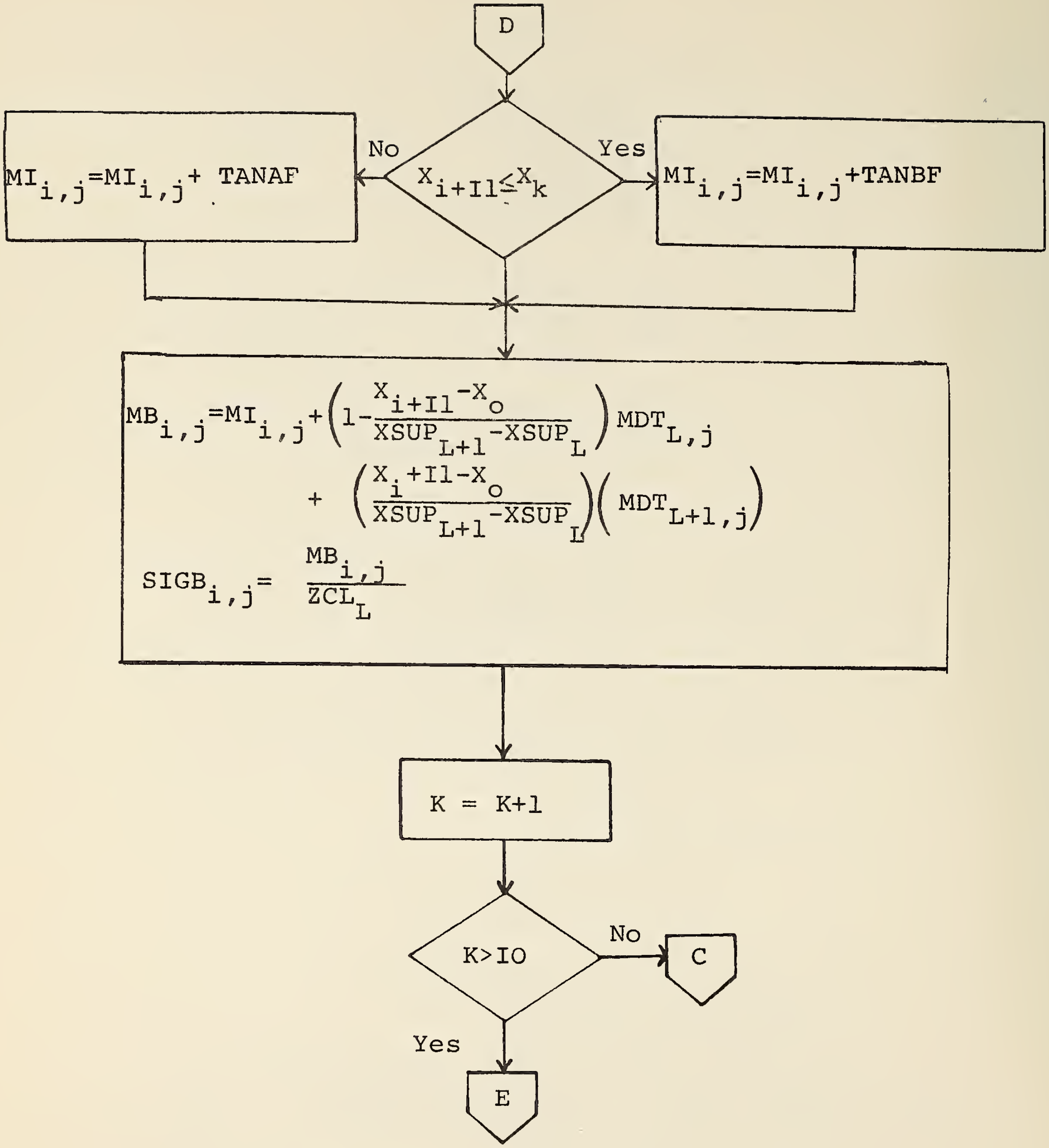

D
No
Yes
$MI_{i,j}=MI_{i,j}+TANAF$
$X_{i+I1}\leq X_k$
$MI_{i,j}=MI_{i,j}+TANBF$
$MB_{i,j}=MI_{i,j}+\left(1-\frac{X_{i+I1}-X_o}{XSUP_{L+1}-XSUP_L}\right)MDT_{L,j}+\left(\frac{X_i+I1-X_o}{XSUP_{L+1}-XSUP_L}\right)\left(MDT_{L+1,j}\right)$
$SIGB_{i,j}=\frac{MB_{i,j}}{ZCL_L}$
K = K+1
K>IO
No
C
Yes
E

Figure 30. ASEMBL Program Flow Chart (Continued)

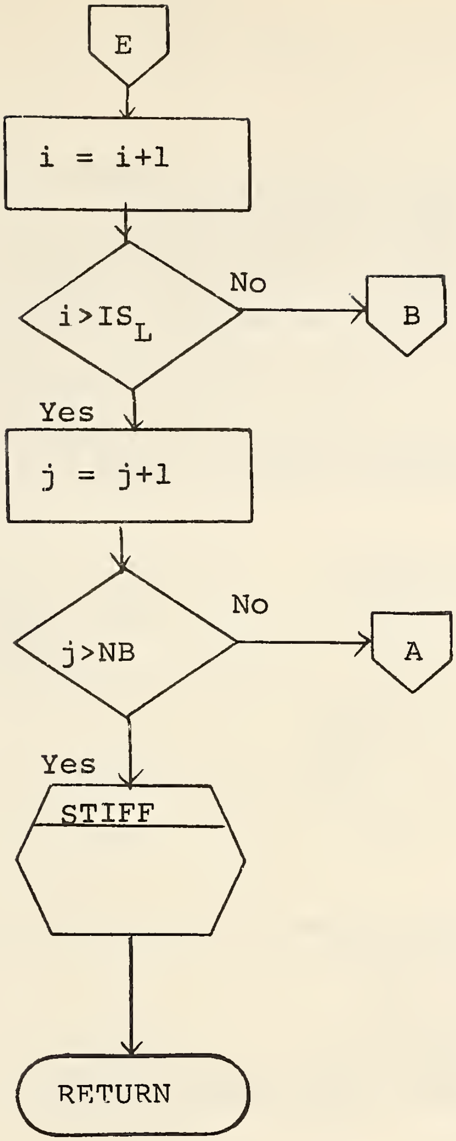


Figure 30. ASEMBL Flow Chart (Continued)

```
0001                  SUBROUTINE ASEMBL(L)
                C
0002                  REAL MI,MB,MDT
                C
0003                  COMMON /BLK2/ DT,       THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     (
                     1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                     2   NXPTS, NDBUG(10),
                     3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                     4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                     6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                     7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
                C
0004                  COMMON /BLK4/ XK(80,80),XKI(80,80), ATEMP(80,80), DTEMP(80)
                     1          ,MI(20), MB(20), SIGB(20,10)
                C
0005                  DIMENSION SK(20,20,10), SKP(10,10)
0006                  EQUIVALENCE (SK(1,1,1),ATEMP(1,1)), (SKP(1,1),ATEMP(1,51))
                C
                C
0007                  I1 = 0
0008                  XO = XSUP(L)
0009                  X1 = XSUP(L+1)
                C
0010                  LM1 = L - 1
                C
0011                  IF (LM1.EQ.0)  GO TO 11
0012                  DO 10 I=1,LM1
0013               10 I1 = I1 + IS(I)
0014               11 IO = I1 + IS(L)
0015                  I1 = I1 + 1
                C
0016                  ISL = IS(L)
                C
                C
0017                  DO 100 J = 1,NB
0018                  DO 16 I=1,20
0019                  MB(I) = 0.
0020                  MI(I)   = 0.
0021               16 CONTINUE
0022                  DO 50 I=I1,IO
0023                  TANTHA = (F(I,J)*(X1-X(I)))/(X1-XO)
0024                  TANTHB = (F(I,J)* X(I))/(X1-XO)
                C
0025                  KK = 0
                C
0026                  DO 20 K=I1,IO
0027                  KK = KK + 1
0028                  IF (X(K).LE.X(I))  GO TO 15
0029                  MI(KK) = MI(KK) + (X1-X(K))*TANTHB
0030                  GO TO 20
                C
0031               15 CONTINUE
0032                  MI(KK) = MI(KK) + X(K)*TANTHA
0033               20 CONTINUE
```

```
   50 CONTINUE
C
      II = 0
      DO 70 I=I1,I0
      II = II + 1
      MB(II)  = MI(II)  + (1.0 -(X(I)      -X0)/(X1-X0))*MDT(L,J)
     1                  + ((X(I)      -X0)/(X1-X0)) * ( MDT(L+1,J))
      SIGB(I,J) = (MB(II) /ZCL(L,J))/144.0
   70 CONTINUE
C
      IF (NDBUG(3).EQ.0)  GO TO 100
      WRITE (IO,1040)
      WRITE (IO,1041)  MB
 1040 FORMAT ('0 MB    ')
 1041 FORMAT (1X,10E13.5)
      WRITE (IO,1044)
      WRITE (IO,1041)  MI
 1044 FORMAT ('0 MI    ')
  100 CONTINUE
C
      IF (NDBUG(3).EQ.0)  GO TO 60
      WRITE (IO,1042)
      WRITE (IO,1043)  SIGB
 1042 FORMAT ('0 SIGB ')
 1043 FORMAT (1X,10E13.5)
   60 CONTINUE
C
      CALL STIFF (L)
C
      RETURN
      END
```

BRGDYN

This subroutine calculates the static and dynamic deflections, the dynamic load factor, and the dynamic bending stress.

Given the nonhomogeneous equation of motion, for forced undamped vibration,

$$[F_k] = [K]\ [\delta_t] + [M\ddot{\delta}_t]$$

where t is the time increment index, that is, time t occurs at t and time t-Δt occurs at t-1, K is the constant assembled stiffness matrix for a span, M is the mass of each element, δ the vertical position of each element, $\dot{\delta}$ the vertical velocity of each element and $\ddot{\delta}$ the vertical acceleration of each element. The positive vertical direction is downward.

In order to minimize computer time a presolved simultaneous solution of the numerical integration equations and the equation of motion has been utilized. A double trapezoidal integration formula was used as a basis, that is,

$$[\delta_t] = [\delta_{t-1}] + \frac{\Delta t}{2}\ [\ddot{\delta}_t] + [\ddot{\delta}_{t-1}]$$

and

$$[\delta_t] = [\delta_{t-1}] + \frac{\Delta t}{2}\ [\dot{\delta}_t] + [\dot{\delta}_{t-1}]$$

We first multiply the flexibility matrix, $[K]^{-1}$, through the equation of motion,

$$[K]^{-1}[F_t] = [\delta_t] + [K]^{-1}[M\ddot{\delta}_t]$$

Solving for

$[\delta_t]$:

$$[\delta_t] = [k]^{-1} \left\{ [F_t] - [M\ddot{\delta}_t] \right\}$$

Substituting in the integration formula

$$[\delta_{t-1}] + \frac{\Delta t}{2} \left\{ [\dot{\delta}_t] + [\dot{\delta}_{t-1}] \right\} = [K]^{-1} \left\{ [F_t] - [M\ddot{\delta}_t] \right\}$$

Solving for

$[\dot{\delta}_t]$:

$$[\dot{\delta}_t] = \frac{2}{\Delta t} \left([K]^{-1} \left\{ [F_t] - [M\ddot{\delta}_t] \right\} - [\delta_{t-1}] \right) - [\delta_{t-1}]$$

Substituting for $[\dot{\delta}_t]$ in the remaining integration formula

$$[\dot{\delta}_{t-1}] + \frac{\Delta t}{2} \left\{ [\ddot{\delta}_t] + [\ddot{\delta}_{t-1}] \right\} = \frac{2}{\Delta t} \left([K]^{-1} \left\{ [F_t] - [M\ddot{\delta}_t] \right\} - [\delta_{t-1}] \right) - [\dot{\delta}_{t-1}]$$

Solving for

$[\ddot{\delta}_t]$:

$$[\ddot{\delta}_t] = \left\{ [I] = \frac{4M}{\Delta t^2} [K]^{-1} \right\}^{-1} \left\{ \frac{4}{\Delta t^2} [K]^{-1} [F_t] - \frac{4}{\Delta t^2} [\delta_{t-1}] - \frac{4}{\Delta t} [\dot{\delta}_{t-1}] - [\ddot{\delta}_{t-1}] \right\}$$

Subsequently, $[\dot{\delta}_t]$ and $[\delta_t]$ are determined from the integration formulas.

Hence, an explicit solution of $[\delta_k]$ exists which is equivalent to an iterative double trapezoidal numerical integration of the nonhomogeneous differential equation of motion.

The dynamic bending stresses are then calculated as

$$\sigma_B(t)_{i,j} = DLF_{i,j} \, \sigma sB_{i,j}$$

where $\sigma sB_{i,j}$ are the previously calculated static bending stresses.

This subroutine does not call any subroutines.

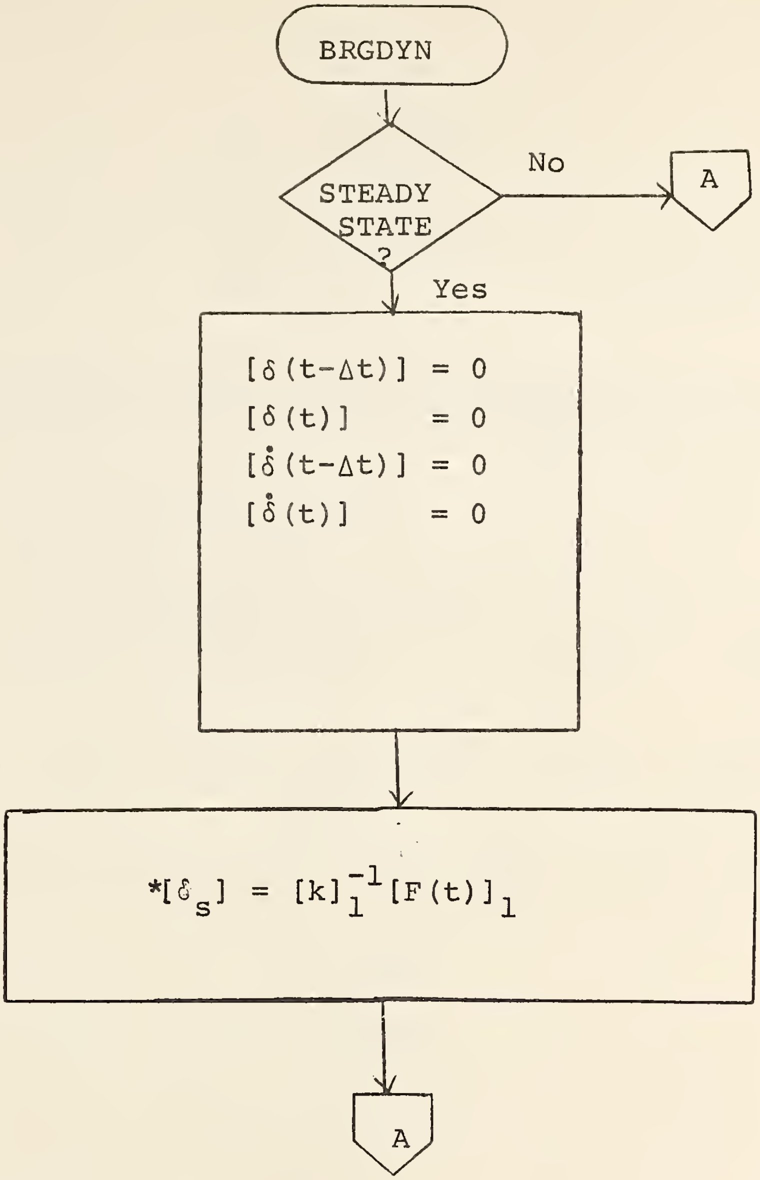
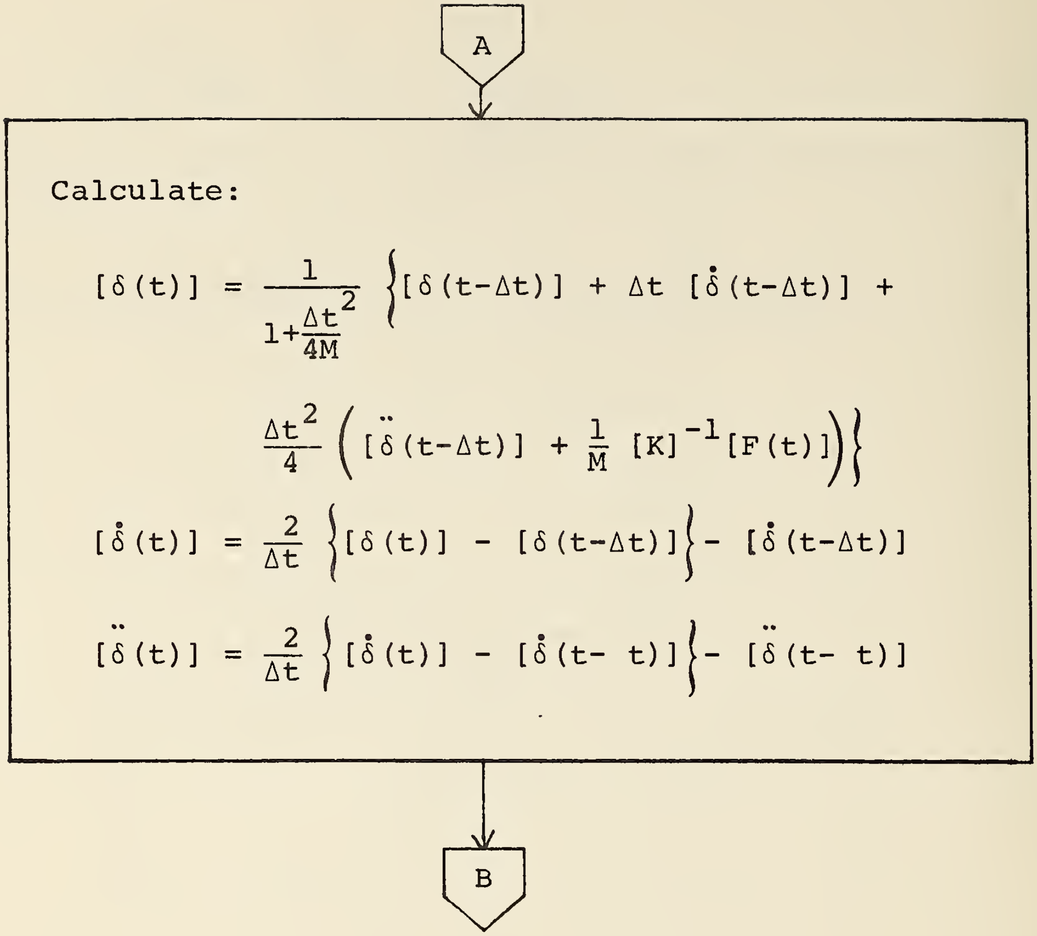

BRGDYN
STEADY STATE ?
No
A
Yes
$[\delta(t-\Delta t)] = 0$
$[\delta(t)] = 0$
$[\dot{\delta}(t-\Delta t)] = 0$
$[\dot{\delta}(t)] = 0$
$*[\delta_s] = [k]_1^{-1}[F(t)]_1$
A

Figure 31. BRGDYN Program Flow Chart



Calculate:

$$[\delta(t)] = \frac{1}{1+\frac{\Delta t^2}{4M}} \left\{[\delta(t-\Delta t)] + \Delta t\,[\dot{\delta}(t-\Delta t)] + \frac{\Delta t^2}{4}\left([\ddot{\delta}(t-\Delta t)] + \frac{1}{M}\,[K]^{-1}[F(t)]\right)\right\}$$

$$[\dot{\delta}(t)] = \frac{2}{\Delta t}\left\{[\delta(t)] - [\delta(t-\Delta t)]\right\} - [\dot{\delta}(t-\Delta t)]$$

$$[\ddot{\delta}(t)] = \frac{2}{\Delta t}\left\{[\dot{\delta}(t)] - [\dot{\delta}(t-\ t)]\right\} - [\ddot{\delta}(t-\ t)]$$

Figure 31. BRGDYN Program Flow Chart (Continued)

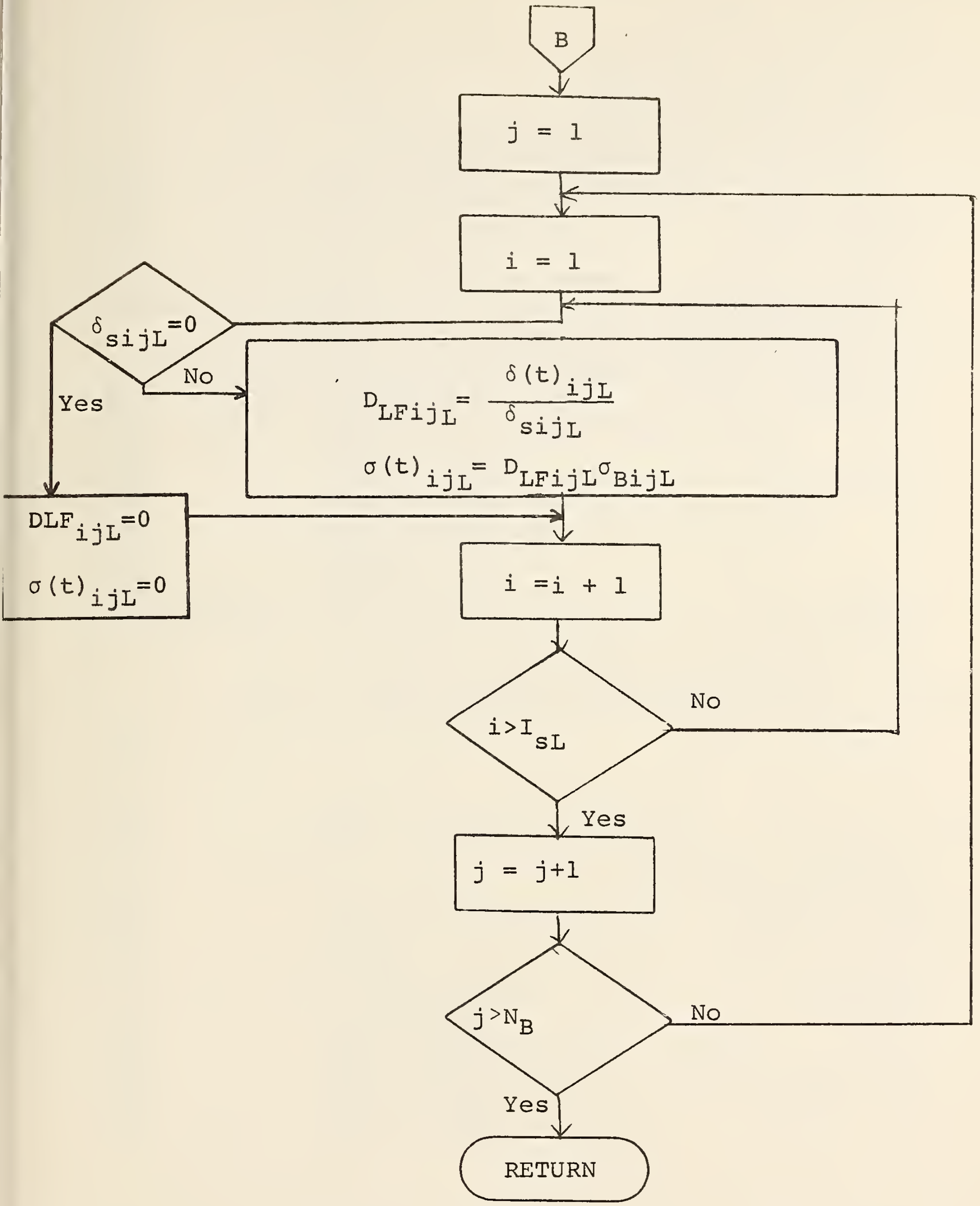


Figure 31. BRGDYN Program Flow Chart (Continued)

```
0001                  SUBROUTINE BRGDYN(L)
              C
              C   THIS SUBROUTINE CALCULATES
              C       U THE STATIC DEFLECTIONS
              C     I THE DYNAMIC DEFLECTIONS
              C     3 THE VELOCITY OF THE DECK
              C     4 THR ACCELERATION OF THE DECK
              C     5 THE DYNAMIC LOAD FACTOR AT EACH POSITION AND AT EACH TIME
              C     6 THE DYNAMIC BENDING STRESS
              C
0002                  REAL MD, MDT
0003                  LOGICAL STEADY
              C
0004                  COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,
                    1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                    2   NXPTS, NDBUG(10),
                    3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                    4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                    6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                    7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
0005                  COMMON /BLK4/ XK(80,80),XKI(80,80), ATEMP(80,80), DTEMP(80)
                    1          ,MI(20), MB(20), SIGB(20,10)
              C
0006                  COMMON /BLK5/ SIG(80),DFLN(80),DFL(80,10),ITIME,KTIME,
                    1     SIGP(120,3,10), SIGC(10,3,10), SIGTHS, DSIG
                    2     , IOPT1, IOPT2, IOPT3, IOPT4, NPTS(10), PTS(3,10)
                    3     , IPTS(3,10)
              C
0007                  COMMON /BLK7/WGT,VEL,DELT,BLEN,NEV,NNL,NBEAM,NP(10),DMM(3),ITAPE2
                    1,SIGOUT(3,10)
              C
0008                  DIMENSION DLF(80),    DFLD(80,10), DFLDD(80,10),
                    1       DFLND(80), DFLNDD(80), DFS(80), FF(80), DUM(80)
0009                  DATA ISTDY,JSTDY/0,0/
              C
0010                  IF(L.EQ.1) ISTDY = 0
0011                  ITER = 3
              C
              C   INITIALIZE FORCE VECTOR FOR L TH SPAN
              C
0012                  IX = 0
0013                  LL = L - 1
0014                  IF(LL.EQ.0) GO TO 11
              C
0015                  DO 1 I=1,LL
0016                1 IX = IX + IS(L)
              C
0017               11 CONTINUE
0018                  IXU = IX + IS(L)
0019                  IX = IX + 1
0020                  II = 1
              C
0021                  DO 2 J=1,NB
0022                  DO 2 I=IX,IXU
```

```
0023                  FF(II) = F(I,J)
0024                2 II = II + 1
            C
0025                  KTER = 0
            C
0026                  IF (NDBUG(9).NE.1)  GO TO 3
            C
0027                  WRITE (IO,1000) L
0028             1000 FORMAT('OFORCE VECTOR FOR SPAN L =',I4)
0029                  WRITE (IO,1001) FF
0030             1001 FORMAT (1X, 10F10.0)
            C
0031                3 NDB = IS(L)*NB
0032                  ISL = IS(L)
            C
            C   IF STEADY STATE CLEAR DEFLECTION STORAGE
            C
0033                  IF (ISS.NE.0)  GO TO 4
0034                  DO 5 I=1,80
0035                  DFS(I) = 0.
0036                  DFLN(I) = 0.
0037                  DFLND(I) = 0.
0038                  DFLNDD(I) = 0.
0039                  DO 5 LL=1,NSI
0040                  DFL(I,LL) = 0.
0041                  DFLD(I,LL) = 0.
0042                5 DFLDD(I,LL) = 0.
0043                  STEADY = .FALSE.
            C
0044                4 CALL MATMUL(XKI, FF, DFS, NDB,  1,  NDB, 80, 80)
            C
            C
            C   SOLVE NON-HOMOGENEOUS DIFFERENTIAL EQUATIONS SIMULTANEOUSLY FOR
            C       EACH GRID OF THE DECK
            C
0045                  C1 = 4./(DT*DT)
0046                  C2 = 4./DT
            C
0047                  DO 6 I=1,NDB
0048                6 DUM(I) = C1*(DFS(I) - DFL(I,L)) - C2*DFLD(I,L) - DFLDD(I,L)
0049                  CALL MATMUL (XK, DUM, DFLNDD, NDB,  1,  NDB, 80, 80)
            C
0050                  DO 7 I=1,NDB
0051                  DFLND(I) = DFLD(I,L) + DT/2. * (DFLNDD(I) + DFLDD(I,L))
0052                  DFLN (I) = DFL(I,L) + DT/2 * (DFLND(I) + DFLD(I,L))
0053                7 CONTINUE
            C
0054                  IF (NDBUG(9).EQ.0)  GO TO 8
0055                  WRITE (IO,2001)
0056                  WRITE (IO,2002) (DFLNDD(I),I=1,NDB)
0057                  WRITE (IO,2003)
0058                  WRITE (IO,2002) (DFLND(I),I=1,NDB)
0059                  WRITE (IO,2005)
0060                  WRITE (IO,1002)
```

```
0061          1002 FORMAT ('0DYNAMIC DEFLECTIONS')
0062               WRITE (IO,2002) (DFLN(I),I=1,NDB)
0063          2001 FORMAT('0DFLNDD')
0064          2003 FORMAT('0DFLND')
0065          2005 FORMAT('0DFLN')
0066          2002 FORMAT(10E13.4)
              C
0067          1003 FORMAT (1X,10E12.3)
0068               WRITE (IO,1004)
0069          1004 FORMAT ('0STATIC DEFLECTIONS')
0070               WRITE (IO,1003)(DFS(I),I=1,NDB)
              C
0071             8 CONTINUE
0072               DO 10 J=1,NB
0073               DO 10 II=1,ISL
0074               I = II + ISL*( J-1)
0075               DFLDD(I,L) = DFLNDD(I)
0076               DFLD(I,L) = DFLND (I)
0077               DFL(I,L) = DFLN(I)
0078               IF (DFS(I).NE.0)  GO TO 20
0079               DFL(I,L) = 0.0
0080               SIG(I) = 0.0
0081               GO TO 10
              C
0082            20 CONTINUE
0083               DLF(I) = DFLN(I)/DFS(I)
0084               SIG(I) = SIGB(II,J) * DLF(I)
0085               IF(ABS(SIG(I)).GT.THRSH)  ISTDY = 1
0086            10 CONTINUE
              C
              C   COLLECT OUTPUT DATA
              C
0087               IF (ITAPE2.GT.0)  GO TO 29
0088               IF(IOPT1.EQ.0.AND.IOPT2.EQ.0.AND.IOPT3.EQ.0)  GO TO 40
0089            29 CONTINUE
              C
0090               DO 30 J=1,NB
0091               NPT = NPTS(J)
0092               IF(NPT.EQ.0)  GO TO 30
              C
0093               DO 25 I=1,NPT
0094               IP = IPTS(I,J)
0095               IF(IP.LT.IX.OR.IP.GT.IXU)  GO TO 25
              C
0096               IPP = IP - IX + 1 + (J-1)*ISL
0097               SIGP(KTIME,I,J) = SIG(IPP)
0098               IF (ITAPE2.GT.0) SIGOUT(I,J)=SIG(IPP)
0099            25 CONTINUE
0100            30 CONTINUE
              C
0101            40 CONTINUE
              C
              C
0102               IF (NDBUG(9).EQ.0)  GO TO 100
```

```
0103                  WRITE (IO,1005)
0104                  WRITE (IO,1003)(DLF(I),I=1,NDB)
0105                  WRITE (IO,1006)
0106                  WRITE (IO,1003)(SIG(I),I=1,NDB)
0107             1005 FORMAT ('ODYNAMIC LOAD FACTOR')
0108             1006 FORMAT ('ODYNAMIC BENDING STRESSES')
            C
            C    CHECK FOR STEADY STATE
            C
0109              100 CONTINUE
0110                  IF (L.NE.NSI)  GO TO 101
0111                  IF (ISTDY.EQ.0.AND.JSTDY.EQ.0)  STEADY = .TRUE.
0112                  JSTDY = ISTDY
0113              101 CONTINUE
0114                  RETURN
0115                  END
```

BROUT

The subroutine BROUT prints out the optional output on an event basis, i.e., Stress vs. Time Graphs and Stress Minimums, Maximums and ranges for specified points. Maximum stress histograms are also updated.

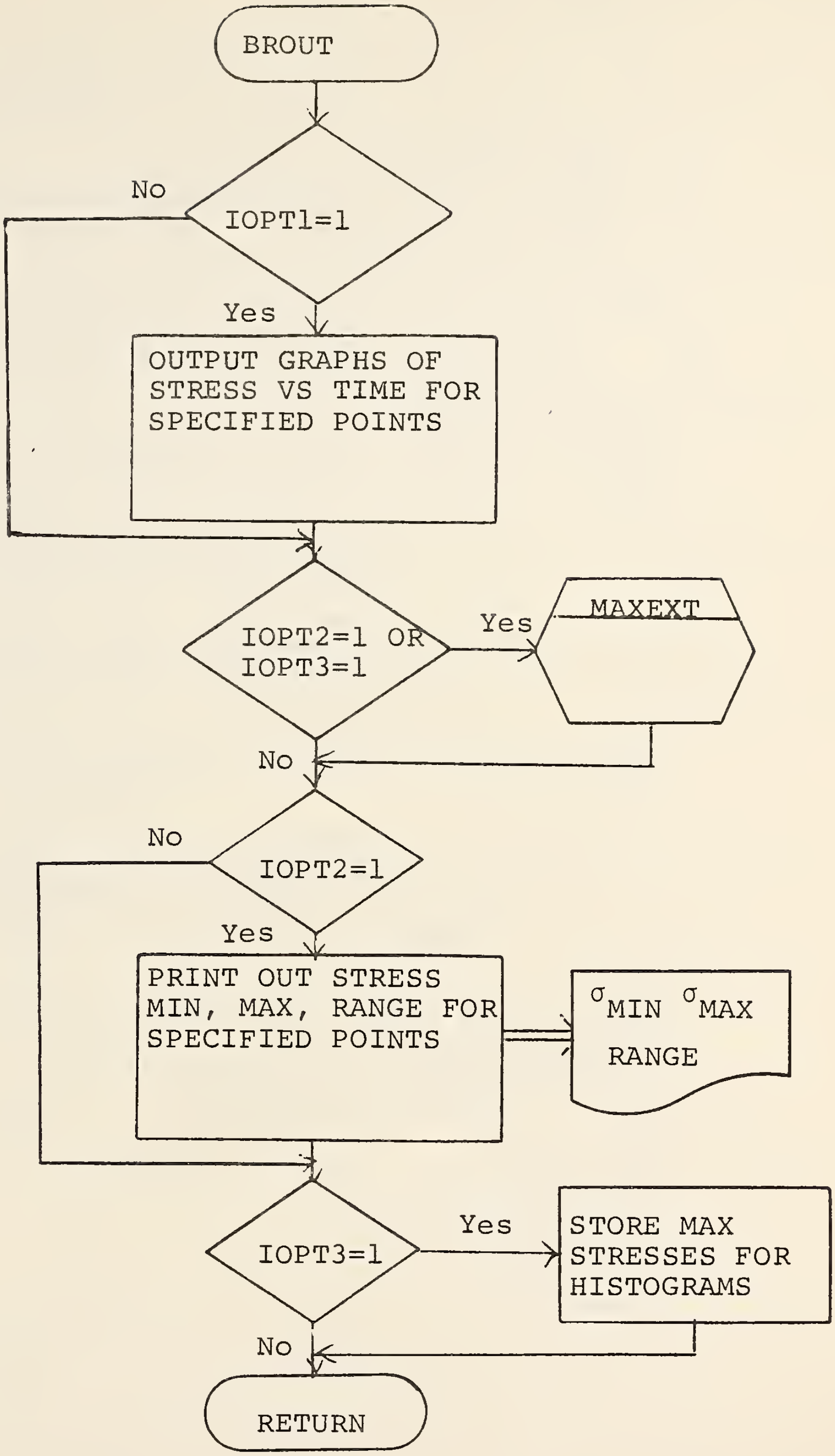

BROUT
IOPT1=1
No
Yes
OUTPUT GRAPHS OF STRESS VS TIME FOR SPECIFIED POINTS
IOPT2=1 OR IOPT3=1
Yes
MAXEXT
No
IOPT2=1
No
Yes
PRINT OUT STRESS MIN, MAX, RANGE FOR SPECIFIED POINTS
σMIN σMAX
RANGE
IOPT3=1
Yes
STORE MAX STRESSES FOR HISTOGRAMS
No
RETURN

Figure 32. BROUT Program Flow Chart

```
0001                SUBROUTINE BROUT
              C
0002                COMMON /BLK1/ IPAGE(120,46), DUM(880)
              C
0003                COMMON /BLK2/ DT,          THRSH,  IE, IEVENT, IN, IO, ISS, ITER,
                   1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                   2   NXPTS, NDBUG(10),
                   3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                   4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                   6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                   7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
              C
0004                COMMON /BLK5/ SIG(80),DFLN(80),DFL(80,10),ITIME,KTIME,
                   1      SIGP(120,3,10), SIGC(10,3,10), SIGTHS, DSIG
                   2      , IOPT1, IOPT2, IOPT3, IOPT4, NPTS(10), PTS(3,10)
                   3      , IPTS(3,10)
              C
0005                DIMENSION TIMEO(13)
0006                DIMENSION NTL(2,46), NDASH(10), KPT(3)
              C
              C
0007                DATA NTL     /4H  5,,4H000-,4*4H     ,4H  4,,4H500-,4*4H     ,
                   1             4H  4,,4H000-,4*4H     ,4H  3,,4H500-,4*4H     ,
                   2             4H  3,,4H000-,4*4H     ,4H  2,,4H500-,4*4H     ,
                   3             4H  2,,4H000-,4*4H     ,4H  1,,4H500-,4*4H     ,
                   4             4H  1,,4H000-,4*4H     ,4H    ,4H500-,4*4H     ,
                   5             4H    ,4H  0-,4*4H     ,4H   -,4H500-,4*4H     ,
                   6             4H  1,,4H000-,4*4H     ,4H -1,,4H500-,4*4H     ,
                   7             4H  2,,4H000-,4*4H     ,4H -2,,4H500-/
              C
0008                DATA NDASH /1H-,1H-,1H-,1H-,1H-,1H-,1H-,1H-,1H-, 1H|/
0009                DATA KPT /1H*,  1H+,  1H./
0010                DATA IBLANK/1H  /
              C
0011                IF (IOPT1.EQ.0)  GO TO 100
              C
              C   OUTPUT GRAPHS OF STRESS VS TIME FOR ALL SPECIFIED POINTS
              C
              C   CALCULATE TIME VALUES
              C
0012                DO 5 J=1,13
0013              5 TIMEO(J) = DT * (ITIME - 1 + (J-1)*10)
0014                IPTEND = (KTIME+9)/10 + 1
0015                IF (NDBUG(8).NE.0)
                   1WRITE(IO,2005) ITIME, KTIME, IPTEND
0016           2005 FORMAT ('OTIME KTIME  IPTEND',3I6)
              C
0017                DO 50 J=1,NB
0018                NPT = NPTS(J)
0019                IF (NPT.EQ.0)  GO TO 50
              C
              C   CLEAR THE GRAPH
              C
0020                DO 10 I=1,46
```

```
      DO 7 K=1,120
      IPAGE(K,I) = IBLANK
    7 CONTINUE
   10 CONTINUE
C
C
    8 DO 9 K=1,120,10
      DO 9 KK = 1,10
    9 IPAGE(KK+K-1,31) = NDASH(KK)
C
C
C  PUT ON THE POINTS
C
      DO 20 I=1,NPT
      DO 20 K=1,KTIME
      XL = 31.0 - ((3.0/ 500.)*(SIGP(K,I,J) -83.0))
      L = XL
      IF (NDBUG(8).NE.0)
     1WRITE (IO,2020) K,I,J,L,SIGP(K,I,J)
 2020 FORMAT(' K I J L',4I6,'     SIGP(K,I,J)', E14.6)
      IF (L.LT.1)  L=1
      IF(L.GT.46)  L=46
      IPAGE(K,L) = KPT(I)
   20 CONTINUE
C
      WRITE (IO,1020) IE,IEVENT,J, (KPT(I),PTS(I,J),I=1,NPT)
 1020 FORMAT (1H1, 'STRESS EVENT',I4,'    LOAD EVENT',
     1                         I4,'    BEAM',I3,'    POINTS',3(4X,A1, F8.3))
C
      WRITE (IO,1021)
      WRITE (IO,1021)
 1021 FORMAT(1H0)
      DO 30 I=1,46
      WRITE (IO,1030)  NTL(1,I),NTL(2,I),(IPAGE(K,I),K=1,120)
 1030 FORMAT (1X,2A4, '|',120A1)
   30 CONTINUE
      WRITE(IO,1035) (TIMEO(I),I=1,IPTEND)
 1035 FORMAT (  'OTIME=',F5.2,12(4X,F6.2))
C
   50 CONTINUE
C
  100 CONTINUE
      IF (IOPT2.EQ.1.OR.IOPT3.EQ.1)  CALL MAXEXT
C
      RETURN
      END
```

MAXEXT

This subroutine extracts points of maxima from a given stress trace. It assumes that the stress values are time wise ordered at a constant interval, Δt. It further assumes that no local noise is induced in the trace. This is predicated on the smooth behavior of the stresses, at a given point, generated by the structural analysis program, as opposed to the type of data acquired from a strain gage. Lastly it assumes that two consecutive zero values of stress imply that the end of the trace has been reached, which is consistent with the steady state criteria used in the structural analysis program. The slope between two points is calculated as

$$S = \frac{\sigma(I) - \sigma(M)}{(I-M)\Delta t}$$

if

$$S \geq 0$$

then

$$M = I,$$
$$I = I + 1$$

and the test repeated until $S<0$.

If $S<0$ then a test is made to determine if

$$|\sigma(I) - \sigma(M)| > \sigma_{\text{Threshold}}$$

Where $\sigma_{\text{Threshold}}$ is established by a user as the minimum decay in stress which identifies a local maximum. If this threshold is not exceeded the next stress value is compared with the most forward value just tested to determine if decay continues, that is, if

$$\sigma(I+1) < \sigma(I)$$

thus I is incremented and the process is repeated. If

$$\sigma(I+1) \geq \sigma(I)$$

then both M and I are updated as shown above.

If the threshold is exceeded the stress class interval index, N, for the i,j histogram is determined from

$$N = \frac{\sigma(M)}{\Delta\sigma} + 1$$

where N is an integer rounding to the class interval containing $\sigma(M)$. The class interval is then incremented for the extracted stress maxima by the previously determined incidence FREQ1, that is

$$\sigma_c(N,i,j) = \sigma_c(N,i,j) + \text{FREQ1}.$$

This subroutine then returns to the calling program.

No subroutines are called by this subroutine.

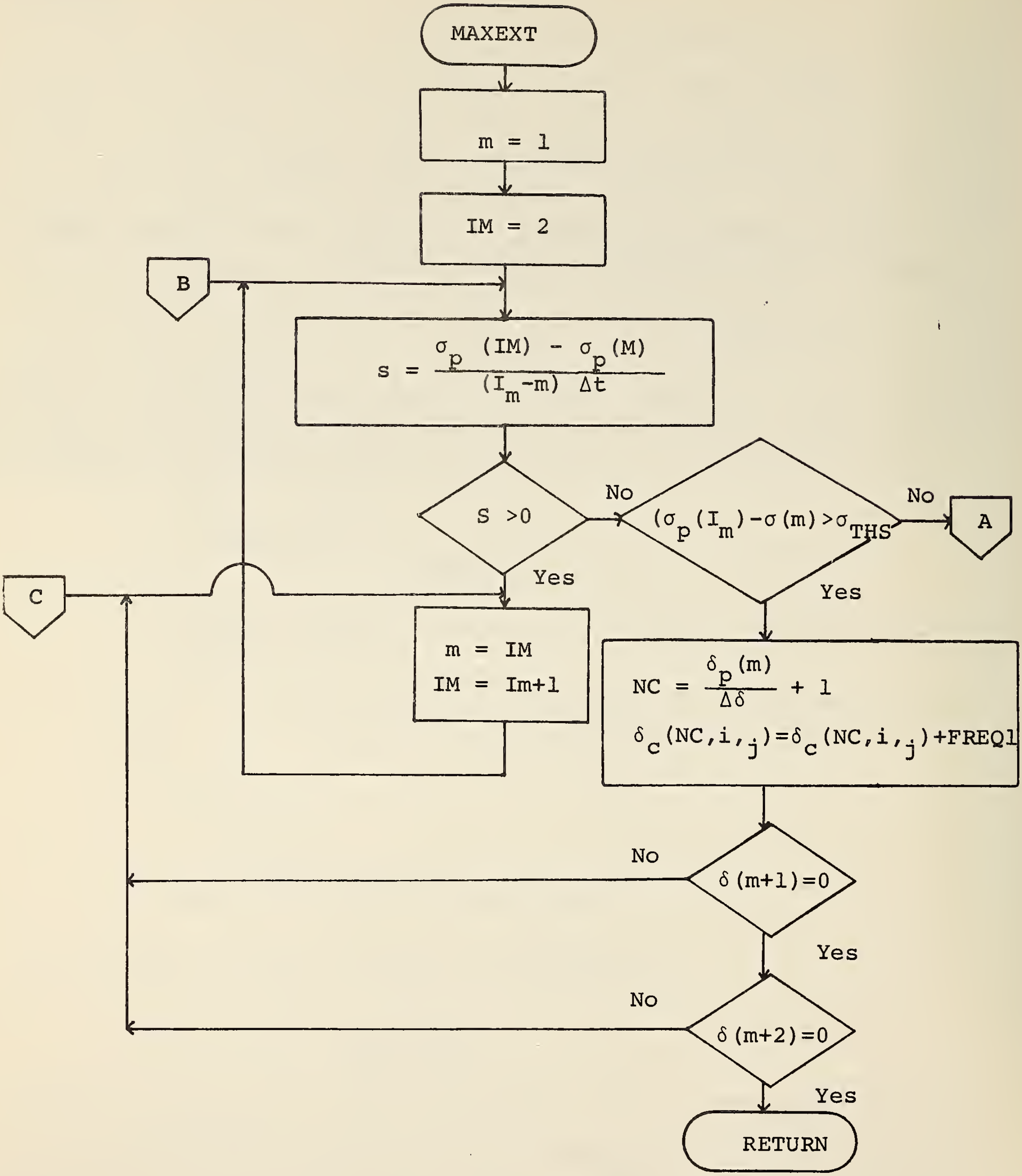

MAXEXT
m = 1
IM = 2
B
$s = \frac{\sigma_p(IM) - \sigma_p(M)}{(I_m - m)\ \Delta t}$
S >0
No
$(\sigma_p(I_m) - \sigma(m) > \sigma_{THS}$
No
A
Yes
C
m = IM
IM = Im+1
Yes
$NC = \frac{\delta_p(m)}{\Delta\delta} + 1$
$\delta_c(NC, i, _j) = \delta_c(NC, i, _j) + FREQ1$
No
$\delta(m+1)=0$
Yes
No
$\delta(m+2)=0$
Yes
RETURN

Figure 33. MAXEXT Program Flow Chart

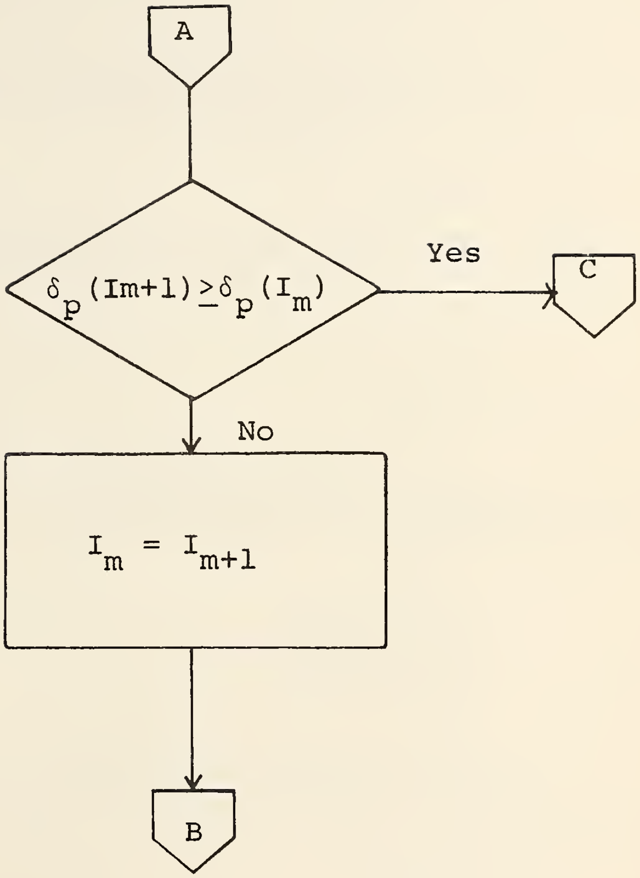


Figure 33. MAXEXT Program Flow Chart (Continued)

```
0001                 SUBROUTINE MAXEXT
               C
0002                 INTEGER SIGC
0003                 COMMON /BLK2/ DT,       THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     0
                    1   NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                    2   NXPTS, NDBUG(10),
                    3   DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                    4   XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                    6   E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                    7   F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
               C
0004                 COMMON /BLK5/ SIG(80),DELN(80),DFL(80,10),ITIME,KTIME,
                    1      SIGP(120,3,10), SIGC(10,3,10), SIGTHS, DSIG
                    2      , IOPT1, IOPT2, IOPT3, IOPT4, NPTS(10), PTS(3,10)
                    3      , IPTS(3,10)
               C
0005                 DIMENSION ITST(3,10), SIGMIN(3,10), TIMMIN(3,10)
               C
               C     THIS SUBROUTINE EXTRACTS THE MAXIMA STRESSES FROM A DIGITAL
               C     STRESS TRACE GIVEN A MINIMUM DROP IN STRESS FROM THE LOCAL MAXIMA
               C
0006                 IF (ITIME.NE.1)  GO TO 6
0007                 DO 5 J=1,10
0008                 DO 5 K=1,3
0009               5 ITST(K,J) = 0
0010               6 CONTINUE
               C
0011                 IF (IOPT2.EQ.0)  GO TO 7
0012                 WRITE (IO,1000) IE, IEVENT
0013            1000 FORMAT (1H1,'STRESS EVENT',I4,'     LOAD EVENT',I4,//)
0014               7 CONTINUE
               C
0015                 DO 100 J=1,NB
0016                 NPT = NPTS(J)
0017                 IF (NPT.EQ.0)  GO TO 100
0018                 DO 95  K=1,NPT
0019                 IF (IOPT2.EQ.0)  GO TO 10
               C
0020                 WRITE (IO,1010) J, PTS(K,J)
0021            1010 FORMAT (1H0,'BEAM NO.', I4, '     POINT',F10.3,//1X,
                    1      'STRESSES: MINIMUM   TIME',10X,'MAXIMUM   TIME',12X,'RANGE')
               C
0022            1012 FORMAT (1X,2(7X,F10.2,F7.3),7X,F10.2)
0023              10 CONTINUE
               C
               C     START THE SEARCH
               C
0024                 IM = 2
0025                 M = 1
0026               2 CONTINUE
0027                 IF (IM.GT.KTIME)  GO TO 90
0028                 S = (SIGP(IM,K,J) - SIGP(M,K,J))
0029                 IF (S.LT.0.)  GO TO 1
0030                 IF (ITST(K,J).NE.0)  GO TO 4
```

```
      IF(S.LT.SIGTHS)   GO TO 15
      SIGMIN(K,J) = SIGP(M,K,J)
      TIMMIN(K,J) = DT*(ITIME-1+M)
      ITST(K,J) = 1
C
    4 CONTINUE
      M = IM
      IM = IM + 1
      GO TO 2
C
    1 CONTINUE
      IF(ITST(K,J).EQ.0)  GO TO 4
      SABS = ABS(SIGP(IM,K,J) - SIGP(M,K,J))
      IF (SABS.GT.SIGTHS)  GO TO 3
C
      IF(SIGP(IM+1,K,J).GE.SIGP(IM,K,J))   GO TO 4
C
   15 CONTINUE
      IM = IM + 1
      GO TO 2
C
C     UPDATE THE APPROPRIATE HISTOGRAM STRESS CLASS INTERVAL
C
    3 NC = SIGP(M,K,J)/DSIG + 1
      IF (NC.GT.10)   NC = 10
      SIGC(NC,K,J) = SIGC(NC,K,J) + 1
C
      IF (IOPT2.EQ.0)  GO TO 80
      SIGMAX = SIGP(M,K,J)
      SIGRNG = SIGMAX - SIGMIN(K,J)
      TIMMAX = DT * (ITIME - 1 + M)
      WRITE (IO,1012) SIGMIN(K,J),TIMMIN(K,J), SIGMAX,TIMMAX,SIGRNG
   80 CONTINUE
C
      ITST(K,J) = 0
      GO TO 4
C
   90 CONTINUE
   95 CONTINUE
  100 CONTINUE
C
      RETURN
      END
```

INCRMT

The INCRMT routine updates the positions of the axle loads

$$XPOSX_i = XPOSX_i + \frac{DXPOSX_i}{NI}$$

and updates the time

$$T = T + DT$$

If the debug output is specified, the axle positions and distance traveled over the load interval are printed out.

No subroutines are called by this routine.

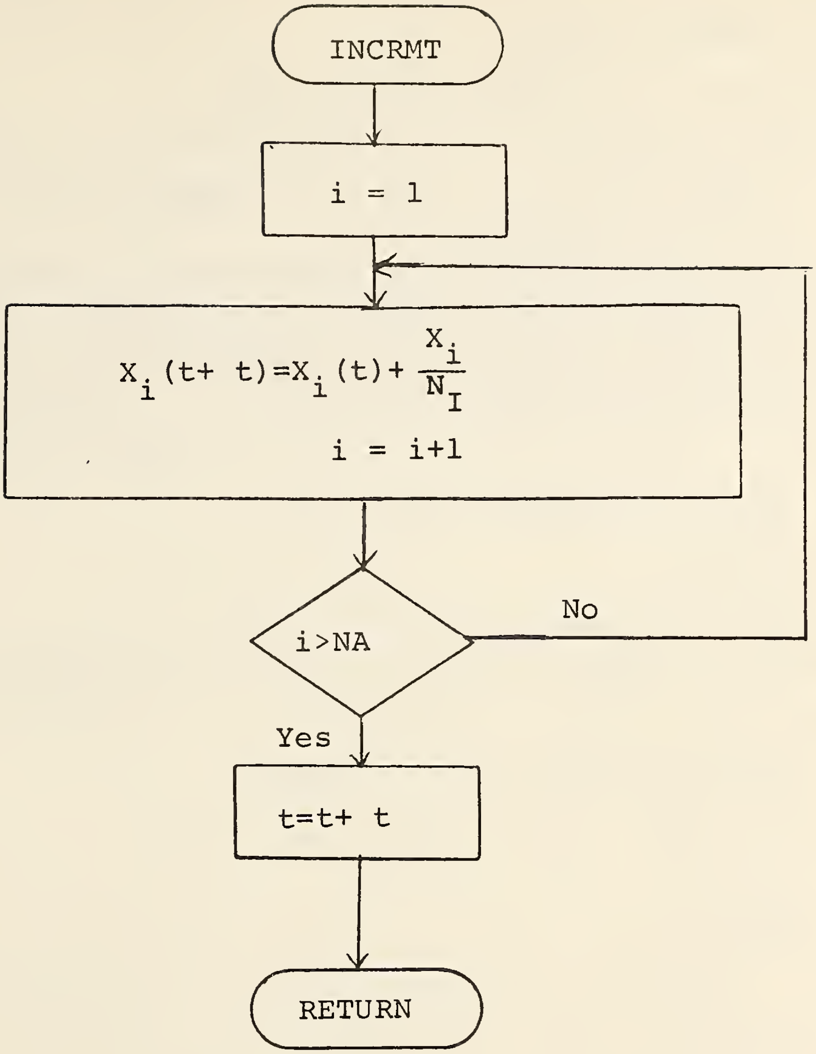


Figure 34. INCRMT Program Flow Chart

```
0001                SUBROUTINE INCRMT
              C
              C THIS SUBROUTINE INCREMENTS THE AXLE LOADS
              C
0002                REAL MD, MDT
              C
0003                COMMON /BLK2/ DT,      THRSH,  IE, IEVENT, IN, IO, ISS, ITER,     (
                  1  NAP, NB, ND, NI, NL, NSI, NSUP, STEADY, T, WEIGHT, XLB, YBCL,
                  2  NXPTS, NDBUG(10),
                  3  DELW(10), DELX(10), EEI(10,10), EIL(10), IS(10), NCS(10),
                  4  XSUP(10), YB(10), YP(4), YCL(10), ZCL(10,10), ZCU(10,10),
                  6  E(3,10), EI(3,10), XCS(3,10), ZBL(3,10), ZBU(3,10),
                  7  F(80,10), MD(11,20,10), MDT(11,10), WP(50,4), X(80),XP(50)
              C
0004                COMMON /BLK3/ SUMHRX, DTL, IEVNTX, NA,     NTRUKX, LAST,
                  1    LTYPE(20), WGTX(20),SPDX(20), LLANE(20), TIMEX(20),
                  2    XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
              C
0005                DO 1 II=1,NA
0006                XPOSX(II)=XPOSX(II)+DXPOSX(II)/NI
0007              1 CONTINUE
0008                T=T+DT
              C
0009                IF (NDBUG(10).EQ.0)  GO TO 30
0010                WRITE (IO,1001) NA,NI
0011           1001 FORMAT('OINCRMT    NO AXLES =',I4,'    NO INCREMENTS =',I4,//,
                  1     '      X            DX')
0012           1020 FORMAT(1X,2F12.4)
0013                DO 20 II=1,NA
0014                WRITE (IO,1020) XPOSX(II),DXPOSX(II)
0015             20 CONTINUE
              C
0016             30 CONTINUE
              C
0017                RETURN
0018                END
```

MATINV

The MATINV subroutine computes the inverse of a general real Matrix A of order N.

The method used is a Gaussian elimination with partial pivoting used to obtain a first estimate, B_0, to the inverse of A. Corrections may be performed, as an option, using the iterative scheme

$$B_{r+1} = B_r(2I-AB_r)$$

where I is the identity matrix and $C_r=2I-AB_r$ is the correction matrix needed to calculate the B_{r+1}th estimate. At each step, the one-norm of C_r-I is computed, i.e.,

$$[C]_1 = \max_i \left| \left(\sum_{j=1}^{n} |c_{ij}| \right) -1 \right|$$

The user actually specifies only a key indicating whether he does or does not want the corrections performed. A maximum of three corrections will be performed since no further improvement can be expected. However, if the condition

$$\left| \left[2I-AB_{r+1}\right]_1 - \left[2I - AB_r\right]_1 \right| \leq 2^{-45} \left[2I-AB_r\right]_1$$

is satisfied, the iteration stops and B_{r+1} is returned as the final inverse. This condition indicates that the machine has done as well as it can in obtaining an inverse and further iterations will not improve the result.

The calling sequence is:

```
CALL MATINV(NDIM,N,A,B,C,D,MAXIT,IERR)
```

where

NDIM -- the formal row dimension of A, B, and C in the calling program (integer).

N -- the order of the matrix to be inverted, $N \leq NDIM$ (integer).

A -- matrix to be inverted, whose row dimension must be NDIM, and whose column dimension must be at least N. This matrix is <u>not</u> destroyed (real).

B -- inverse of A, whose row dimension must be NDIM and whose column dimension must be at least N (real).

C -- work matrix whose row dimension must be NDIM and whose column dimension must be at least N (real). Note that if corrections are performed, C contains 2I-AB where B is the returned inverse.

D -- vector of working storage of at least N locations (real).

MAXIT -- correction indicator (integer).

= 0 -- no corrections

≠ 0 -- maximum of three corrections

On return, MAXIT contains the number of corrections performed.

IERR -- error return (integer).

0 -- normal return

1 -- zero pivot found, matrix singular

2 -- norm of correction matrix greater than 1, matrix may not be convergent, return last inverse as estimate. The user should carefully check the inverse for this error return.

No subroutines are used by this routine.

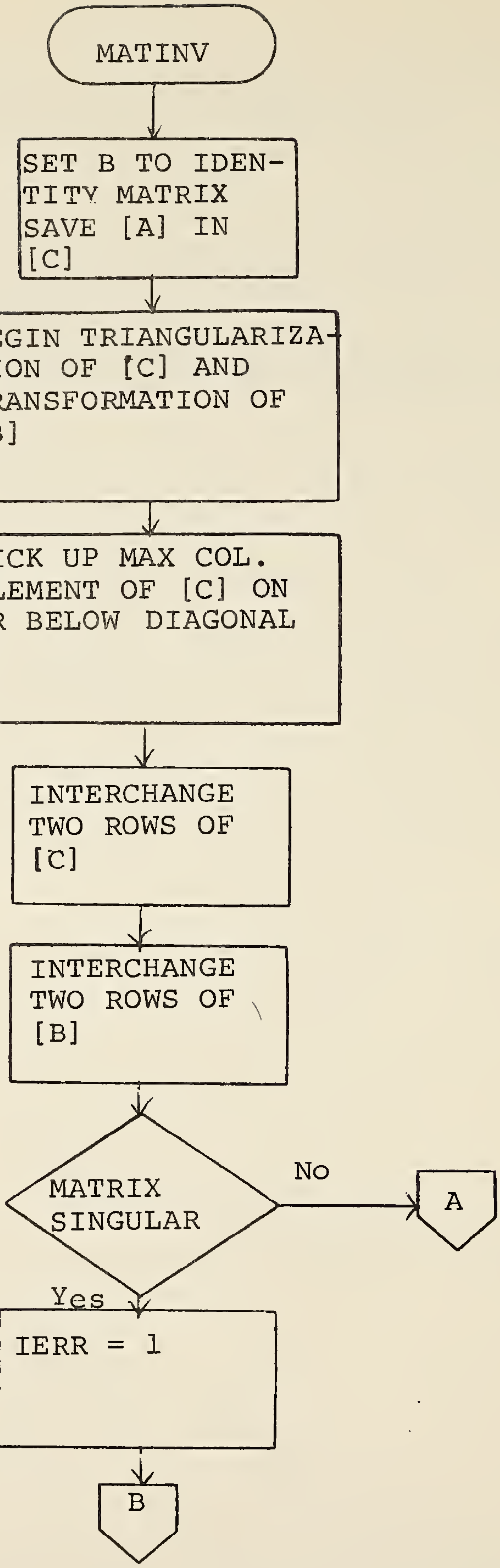

MATINV
SET B TO IDENTITY MATRIX SAVE [A] IN [C]
BEGIN TRIANGULARIZATION OF [C] AND TRANSFORMATION OF [B]
PICK UP MAX COL. ELEMENT OF [C] ON OR BELOW DIAGONAL
INTERCHANGE TWO ROWS OF [C]
INTERCHANGE TWO ROWS OF [B]
MATRIX SINGULAR
No
A
Yes
IERR = 1
B

Figure 35. MATINV Program Flow Chart

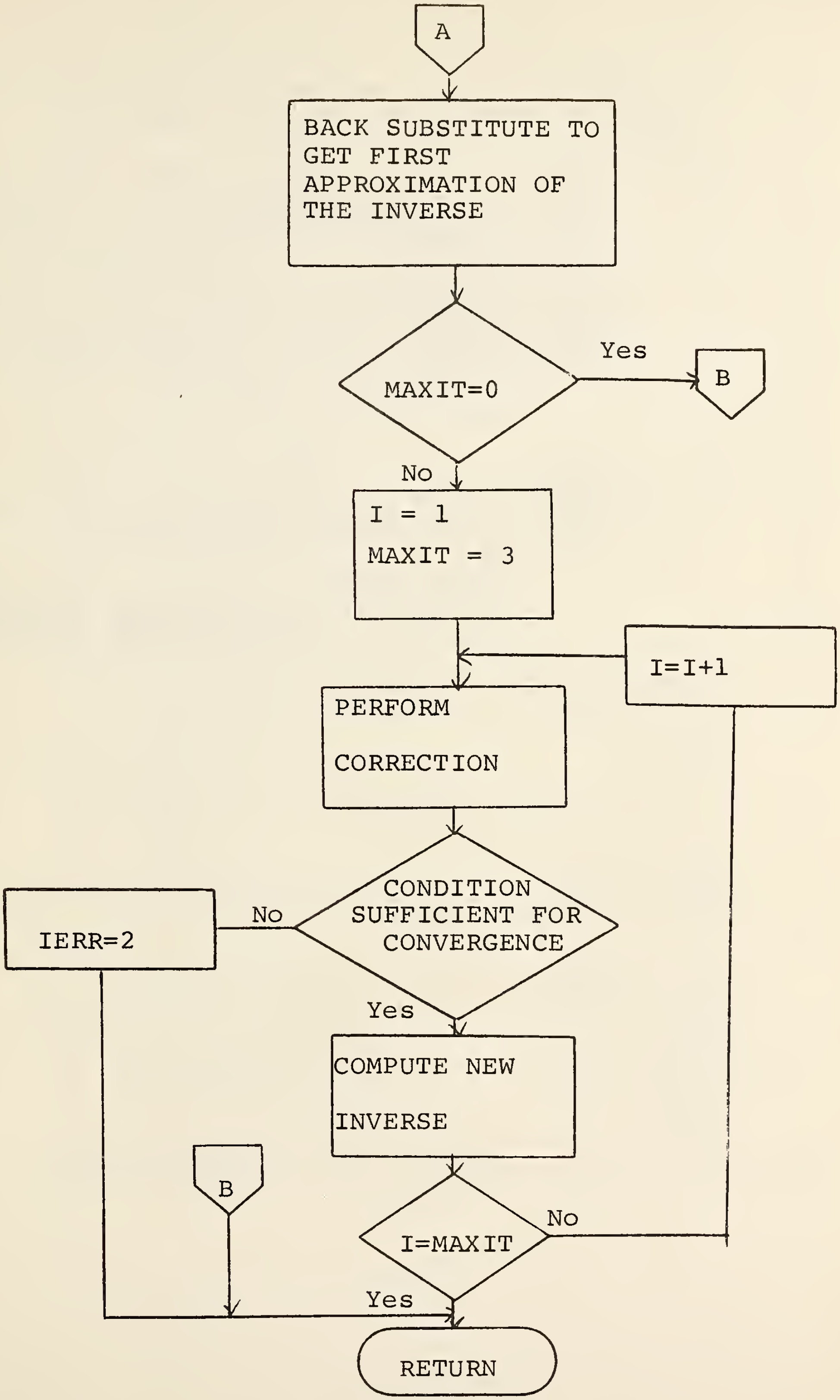


Figure 35. MATINV Program Flow Chart (Continued)

```
0001              SUBROUTINE MATINV (NDIM,N,A,B,C,D,MAXIT,IERR)
            C
            C   SUBROUTINE TO COMPUTE THE ITERATIVELY REFINED INVERSE OF A GENERAL
            C      REA  MATRIX
            C      REAL MATRIX.  TGE FIRST APPROXIMATION IS OBTAINED BY STANDARD
            C      GAUSSIAN ELIMINATION WITH PARTIAL PIVOTING
            C
            C   INPUT PARAMETERS
            C      NDIM - THE FORMAL ROW DIMENSION OF A IN THE CALLING PROGRAM
            C      N    - SIZE OF MATRIX TO BE INVERTED
            C      A    - MATRIX TO BE INVERTED, DIMENSION AT LEAST A(NDIM,N).
            C             A IS NOT DESTROYED
            C      B    - INVERSE OF A, DIMENSIONED AT LEAST B(NDIM,N) IN THE
            C             CALLING PROGRAM
            C      C    - WORK MATRIX, DIMENSIONED AT LEAST C(NDIM,N) IN THE
            C             CALLING PROGRAM
            C      D    - VECTOR OF WORKING STORAGE OF LENGTH AT LEAST N
            C      MAXIT- ITERATION KEY
            C                  0 - NO ITERATION
            C                  1 - THREE ITERATIONS
            C
            C   OUTPUT PARAMETERS
            C      B    - INVERSE OF A
            C      MAXIT- NUMBER OF ITERATIONS PERFORMED. IF LESS THAN THREE
            C             IMPLIES NORM OF RESIDUAL MATRIX CONVERGED
            C      IERR - ERROR RETURN
            C                  0 - NORMAL RETURN
            C                  1 - MATRIX SINGULAR (ZERO PIVOT ELEMENT FOUND)
            C                  2 - NORM NOT LESS THAN ONE.  ERROR MATRIX MAY NOT BE
            C                      CONVERGENT, DO NOT PERFORM ANY MORE REFINEMENTS
            C
0002              DIMENSION A(NDIM,1), B(NDIM,1), C(NDIM,1), D(1)
0003              REAL*8 DP1,DP2
            C
0004              NM = N - 1
0005              IERR = 0
            C
            C   SET IDENTITY MATRIX, SAVE A
            C
0006              DO 5 I=1,NM
0007              IJ = I + 1
0008              C(I,I) = A(I,I)
0009              B(I,I) = 1.
            C
0010              DO 5 J=IJ,N
0011              C(I,J) = A(I,J)
0012              C(J,I) = A(J,I)
0013              B(I,J) = 0.
0014            5 B(J,I) = 0.
            C
0015              C(N,N) = A(N,N)
0016              B(N,N) = 1.
            C
            C   BEGIN TRIANGULARIZATION OF C AND TRANSFORMATION OF B
```

```
                 C
0017                  DO 100 J=1,NM
0018                  K = J
0019                  T = ABS(C(J,J))
0020                  LL = J + 1
                 C
                 C    PICK UP MAX COL ELEMENT ON OR BELOW DIAGONAL
                 C
0021                  DO 20 I=LL,N
0022                  P = ABS(C(I,J))
0023                  IF (T.GE.P)  GO TO 20
0024                  T = P
0025                  K = I
0026               20 CONTINUE
0027                  IF (K.EQ.J)  GO TO 35
                 C
                 C    INTERCHANGE TWO ROWS OF C
                 C
0028                  DO 25 I=J,N
0029                  T = C(J,I)
0030                  C(J,I) = C(K,I)
0031               25 C(K,I) = T
                 C
                 C    INTERCHANGE TWO ROWS OF B
                 C
0032                  DO 30 I=1,N
0033                  T = B(J,I)
0034                  B(J,I) = B(K,I)
0035               30 B(K,I) = T
0036               35 DP1 = C(J,J)
                 C
                 C    CHECK FOR SINGULARITY
                 C
0037                  IF (DP1.EQ.0.D0)  GO TO 340
0038                  DO 100 I=J,NM
0039                  K = I + 1
                 C
                 C    DP2/DP1 IS THE MULTIPLIER
                 C
0040                  DP2 = C(K,J)
                 C
0041                  DO 95 II=LL,N
0042               95 C(K,II) = C(K,II) - (DP2*C(J,II))/DP1
                 C
0043                  DO 98 II=1,N
0044               98 B(K,II) = B(K,II) - (DP2*B(J,II))/DP1
                 C
0045              100 C(K,J) = DP2/DP1
                 C
                 C    END TRIANGULARIZATION OF C AND TRANSFORMATION OF B
                 C    CHECK FOR SINGULARITY
                 C
0046                  IF (C(N,N).EQ.0.)  GO TO 340
                 C
```

```
               C   BEGIN BACK SUBSTITUTION
               C
0047                 DO 150 II=1,N
0048                 D(N) = B(N,II)/C(N,N)
               C
0049                 DO 140 I=1,NM
0050                 DP1 = 0.D0
0051                 K = N -I
0052                 L = K + 1
               C
0053                 DO 130 J=L,N
0054                 DP2 = D(J)
0055             130 DP1 = DP1 + DP2*C(K,J)
               C
0056                 DP2 = C(K,K)
0057             140 D(K) = (B(K,II) - DP1)/DP2
               C
0058                 DO 150 J=1,N
0059             150 B(J,II) = D(J)
               C
               C   END BACK SUBSTITUTION
               C   WE NOW HAVE FIRST APPROXIMATION OF THE INVERSE
               C
0060                 IF (MAXIT.EQ.0)  GO TO 350
               C
0061                 OLDEL = 0.0
0062                 MAXIT = 3
               C
               C   PERFORM THREE CORRECTIONS
               C
0063                 DO 320 KOUNT=1,MAXIT
               C
0064                 DO 240 I=1,N
0065                 DO 240 J=1,N
0066                 DP1 = 0.
               C
0067                 DO 230 K=1,N
0068                 DP2 = A(I,K)
0069             230 DP1 = DP1 + DP2*B(K,J)
               C
0070                 IF (I.NE.J)  GO TO 240
0071                 DP1 = DP1 - 2.
0072             240 C(I,J) = -DP1
               C
               C   NOW C = (2*I - A*B)
               C
0073                 DEL = 0.
0074                 DO 260 I=1,N
0075                 DP1 = 0.
               C
               C   COMPUTE ONE-NORM (MAX ROW SUM)
               C
0076                 DO 250 J=1,N
0077             250 DP1 = DP1 + ABS(C(I,J))
```

```
C
      TEMP = DP1 -1.
      IF (ABS(TEMP).LE.DEL)  GO TO 260
      DEL = ABS(TEMP)
  260 CONTINUE
C
C   CHECK SUFFICIENT CONDITION FO R CONVERGENCE
C
      IF (DEL.LT.1.)  GO TO 270
      IERR = 2
      WRITE (IO,1003) IERR
 1003 FORMAT('0ERROR ',I4,'MATRIX INVERT   - NORM NOT FOUND. ERROR ',
     1    'MATRIX  MAY NOT BE CONVERGENT')
      GO TO 350
C
C   CHECK FOR NORM CONVERGENCE EPS = 2**(-46)
C
  270 IF (ABS(DEL-OLDEL).GT.(.142E-13) *OLDEL)  GO TO 280
      LL = KOUNT - 1
      GO TO 345
C
C   COMPUTE NEW INVERSE
C
  280 DO 310 I=1,N
      DO 300 J=1,N
      DP1 = 0.
C
      DO 290 K=1,N
      DP2 = B(I,K)
  290 DP1 = DP1 + DP2*C(K,J)
C
  300 D(J) = DP1
C
      DO 310  J=1,N
  310 B(I,J) = D(J)
C
  320 OLDEL = DEL
      GO TO 350
C
  340 IERR = 1
      WRITE (IO,1002) IERR
 1002 FORMAT('0ERROR ',I4,'MARRIX INVERT   - MATRIX SINGULAR')
      GO TO 350
C
  345 MAXIT = LL
C
  350 RETURN
      END
```

MATMUL

The MATMUL subroutine computes the product of the two conformal matrices A (L,N) and B (N,M) and stores the product in the matrix C (L,M). Calling sequence is: CALL MATMUL (A,B,C,L,M,N, LL,NN) where LL is the formal dimension of arrays A and C and NN is the formal dimension of array B.

No subroutines are called by this routine.

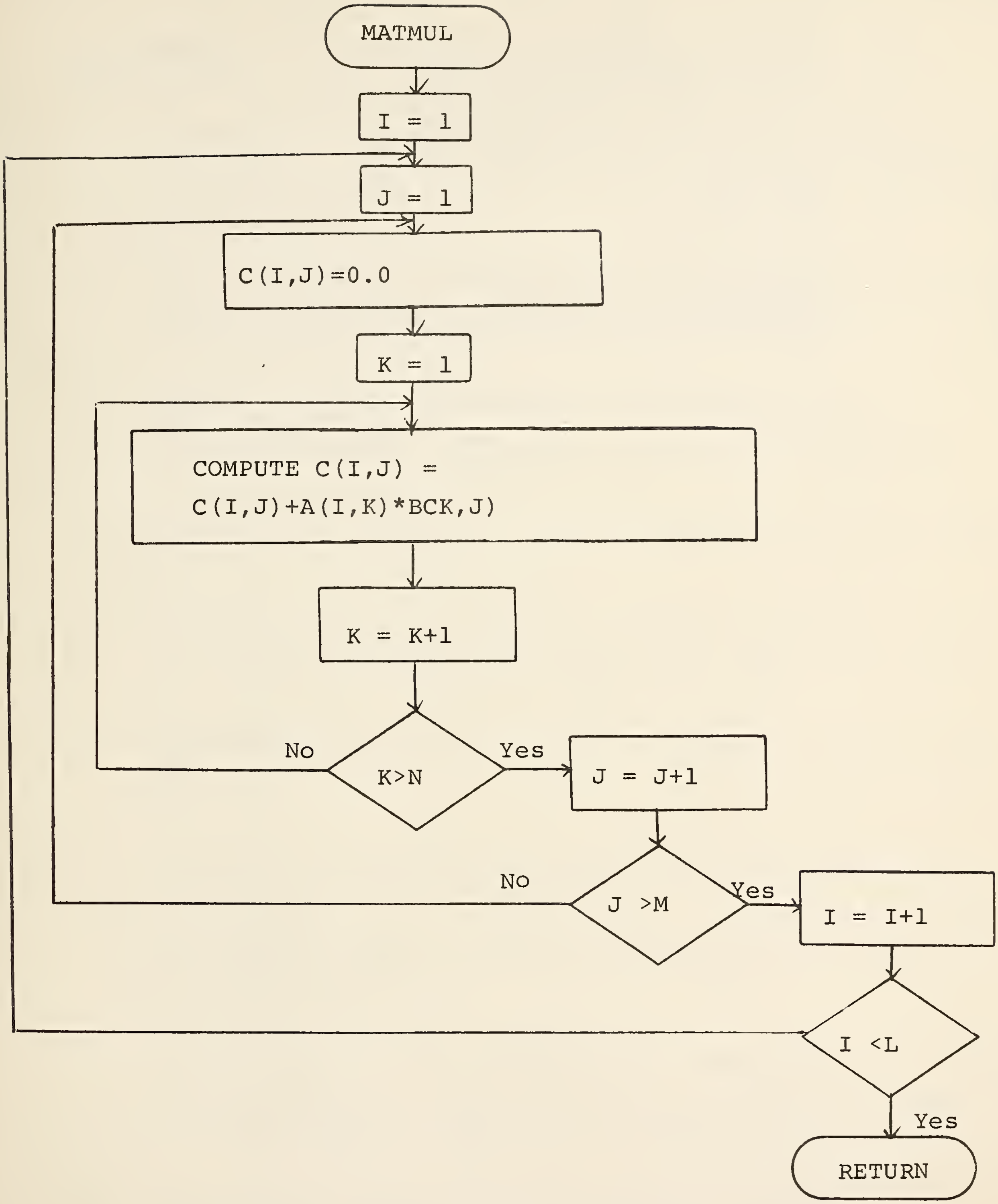


Figure 36. MATMUL Program Flow Chart

```
0001                SUBROUTINE MATMUL (A, B, C, L, M, N, LL, NN)
              C
              C   COMPUTES THE ELEMENTS OF C(L,M) AS A REAULT OF THE PRODUCT OF
              C      A(L,N) AND B(N,M) IN THAT ORDER.
              C   LL IS THE FORMAL DIMENSION OF ARRAYS A AND C
              C   NN IS THE FORMAL DIMENSION OF ARRAY B
              C
0002                DIMENSION A(LL,1), B(NN,1), C(LL,1)
              C
0003                DO 5 I=1,L
0004                DO 5 J=1,M
0005                C(I,J) = 0.0
              C
0006                DO 5 K=1,N
0007                C(I,J) = C(I,J) +A(I,K)*B(K,J)
0008              5 CONTINUE
0009                RETURN
0010                END
```

BIBLIOGRAPHY

1. "Forecasting of Heavy Loading Patterns on Highway Bridges," H. Bissell, et al, Federal Clearing House No. PB193119, Kelly Scientific Corp.

2. "The Finite Element Method," by O. C. Zienkiewicz, Y. K. Cheung, McGraw-Hill Publishing Company

3. "Theory of Plates and Shells," by Timoshenko and Woinowsky-Krieger, McGraw-Hill Book Co. 2nd edition

4. "Strength of Materials," by Timoshenko-Young, McGraw-Hill Book Co.

5. "Structural Design for Dynamic Loads," Charles H. Norris et al (M.I.T.), McGraw-Hill Book Co., 1959

6. "Mechanical Vibrations," by W. T. Thomson, Prentice Hall, Inc.

7. "Analysis of Orthotropic Folded Plates with Eccentric Stiffeners," by K. J. William and A. C. Scordelis, Report No. SESM70-2, Dept. of Civil Engineering, University of California, Berkeley

8. "Theoretical Studies of Bridge Deck Behavior," by Graham H. Powell and Ian G. Buckle, Report No. UCSESM70-7, Dept. of Civil Engineering, University of California, Berkeley

9. "Tabulation of Dynamic Strain Data on a Girder Slab Bridge Structure During Seven Continuous Days," by H. D. Sartwell, C. P. Heins, Jr., Report No. 31, Civil Engineering Dept., University of Maryland at College Park

10. "Study of Truck Weights and the Corresponding Induced Bridge Girder Stresses," by R. L. Khosa, C. P. Heins, Jr., Report No. 40, Civil Engineering Dept., University of Maryland at College Park

11. "Tabulation of Dynamic Strain Data on a Three-Span Continuous Bridge Structure," by A. D. Sartwell, C. P. Heins, Report No. 33, Civil Engineering Dept., University of Maryland at College Park

12. "Loading History of a Highway Bridge - Comparison of Stress Range Histograms" by C. F. Galambos, C. P. Heins, Public Roads, Vol. 36, No. 9, U.S. D.O.T., Washington, D.C.

13. "Acquisition of Loading History Data on Highway Bridges" by C. F. Galambos and W. L. Armstrong, Public Roads, Vol. 35, No. 8, U.S. D.O.T., Washington, D.C.

14. "A Loading History Study of Two Highway Bridges in Virginia" by Wallace T. McKeel, Jr., Charles E. Maddox, Jr., Henry L. Kinnier and Charles F. Galambos, Virginia Highway Research Council, Dept. of Highway and University of Virginia VHRC71-Rpg, 1971

15. "Fracture Toughness Testing and its Applications," ASTM NASA, Special Technical Publication No. 381, Symposium publication at 67th Annual Meeting, 1964

16. "Tabulation of 24 Hour Dynamics Strain Data on Four Simple Span Girder-Slab Bridge Structures," C. P. Heins, Jr., et al, Report No. 29, Civil Engineering Dept., University of Maryland at College Park

17. "Prospects for Urban Transit, " Federal Clearing House No. PBI93355, Charles River Associates, Inc.

18. "The Selection of Improvement Programs on Intercity Highways," Contract Report DOT-OS-A9-060, Charles River Associates, Inc.

19. "Reliability Design for Fatigue under Random Loading" by R. C. Garson, F. Moses, Case Western Reserve University, Ohio Research Study No. 14218, Aug. 1972

GPO 873-603

TE 662 .A3
no. FHWA-RD-
73-44

BORROWE

Form DOT F 17
FORMERLY FORM

DOT LIBRARY
00054405